Un Système de Classification Supervisée à Base de Règles Implicatives

Lavinia Darlea

▶ **To cite this version:**

Lavinia Darlea. Un Système de Classification Supervisée à Base de Règles Implicatives. Intelligence artificielle [cs.AI]. Université de Savoie, 2010. Français. NNT : . tel-01222735

HAL Id: tel-01222735
https://hal.science/tel-01222735

Submitted on 30 Oct 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

THÈSE

pour obtenir

le grade de docteur

Université de Savoie

Spécialité : *Electronique – Electrotechnique – Automatique*

et

Universitatea "Politehnica" Bucureşti

Spécialité : *Electronique – Télécommunications – Technologie de l'Information*

Georgiana–Lavinia DÂRLEA

Un Système de Classification Supervisée à Base de Règles Implicatives

Soutenue le 4 février 2010 devant le jury composé de :

M. Teodor PETRESCU	
M. Cristian GRAVA	Rapporteur
Mme. Anne LAURENT	Rapporteur
M. Vasile BUZULOIU	Directeur de thèse
Mme. Sylvie GALICHET	Directeur de thèse
M. Lionel VALET	Co–Directeur de thèse

Thèse préparée au sein du Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance (Annecy) et du Laboratoire d'Analyse et Traitement des Images (Bucarest)

THÈSE

pour obtenir

le grade de docteur

Université de Savoie

Spécialité : *Electronique – Electrotechnique – Automatique*

et

Universitatea "Politehnica" Bucureşti

Spécialité : *Electronique – Télécommunications – Technologie de l'Information*

Georgiana–Lavinia DÂRLEA

Un Système de Classification Supervisée à Base de Règles Implicatives

Soutenue le 4 février 2010 devant le jury composé de :

M. Teodor PETRESCU	
M. Cristian GRAVA	Rapporteur
Mme. Anne LAURENT	Rapporteur
M. Vasile BUZULOIU	Directeur de thèse
Mme. Sylvie GALICHET	Directeur de thèse
M. Lionel VALET	Co–Directeur de thèse

Thèse préparée au sein du Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance (Annecy) et du Laboratoire d'Analyse et Traitement des Images (Bucarest)

Table des matières

Remerciements

Les travaux présentés dans cette thèse ont été effectués en cotutelle au sein du Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance (LISTIC, Annecy) et du Laboratoire d'Analyse et Traitement des Images (LAPI, Bucarest). Ainsi, je souhaiterais remercier très chaleureusement toutes les personnes qui ont contribué à ce que ma thèse se déroule dans d'excellentes conditions.

Tout d'abord, je tiens à remercier particulièrement Monsieur Philippe BOLON, directeur du LISTIC, et Monsieur Vasile BUZULOIU qui ont eu la gentillesse de m'accueillir au sein de leurs laboratoires.

J'exprime sincèrement ma plus grande gratitude à Madame Sylvie GALICHET, professeur à l'Université de Savoie, pour la qualité de son encadrement, sa disponibilité de tous les instants et pour son aide constante tout au long de cette thèse. Je remercie également très vivement Monsieur Lionel VALET, maître de conférences à l'Université de Savoie, pour ses conseils très précieux, sa patience et la confiance qu'il m'a accordée.

J'exprime ma plus grande reconnaissance à Monsieur le professeur Teodor PETRESCU pour l'honneur qu'il me fait d'accepter de présider ce jury.

Je remercie vivement Monsieur Cristian GRAVA, professeur à l'Université d'Oradea et Madame Anne Laurent, maître de conférence au Laboratoire d'Informatique de Robotique et de Micoélectronique de Montpellier (LIRMM), d'avoir accepté d'être rapporteurs de ma thèse.

I would also like to thank all the colleagues that I had the chance to meet in the last years, during my stay at CERN. First of all I would like to thank them all for their patience with regard to my work on this thesis. And much more important than that, I would like to give them a big hug for just being great friends : Silvia, Matei, Stefan, Lucian and Adina, Costin, Rune, John Erik, Liviu, Sergio, Irina, Dan, Ivan, Anja, Claire... And, of course, the ones that have been my supervisors and that have always had the kindness of supporting me during my struggles between the work and the PhD thesis : Brian, Giovanna and Marc.

Un mare "multumesc" îi adresez domnului Vasile BUZULOIU, caruia ii datorez întregul meu parcurs de până acum. Ţin sa îi mulţumesc pentru încrederea acordată, pentru sprijinul ştiinţific pe care l-am găsit întotdeauna la domnia sa, pentru delicateţea şi tactul cu care a ştiut întotdeauna sa îmi ghideze alegerile cele mai dificile. Aş vrea sa le mulţumesc tuturor membrilor LAPI, în special domnilor Mihai Ciuc şi Constantin Vertan, care mi-au îndrumat primii paşi spre un parcurs post–universitar. De asemenea, ţin sa îi mulţumesc lui Bogdan pentru sprijinul pe care mi l-a acordat atât pe plan ştiinţific cât şi personal, în momente cheie ale activităţii legate de această teză.

Biensur, toute ma reconaissace va vers deux grands amies que j'ai eu la chance de rencontrer au cours de la thèse : Azadeh et Sylvie. Et vers Laurent.

Enfin, je terminerai en remerciant tous les chercheurs, doctorants, stagiaires, techniciens et administratifs du LISTIC, de l'Université de Savoie, du LAPI et de l'Université Polytechnique de Bucarest.

Introduction générale

La classification de données est une problèmatique très connue dans le monde scientifique, car elle est à l'origine de nombreuses applications. Aujourd'hui on la rencontre dans des domaines très variés, tel que le domaine médical, industriel, de la sécurité, etc. Pour tenter de répondre à cette problèmatique, le monde scientifique et académique a proposé de nombreux systèmes essayant de prendre en compte au mieux les caractéristiques spécifiques des applications (gestion de l'incertitude, du manque d'information, des situations conflictuelles, etc). Les systèmes de classification existants s'appuient sur divers formalismes mathématiques pour tenter de représenter et manipuler au mieux les données ainsi que la relation qui les lie aux classess à identifier. Ce document en proposera d'ailleurs une classification suivant trois principaux groupes.

Même si de nombreux systèmes sont disponibles aujourd'hui, une solution complète et globale permettant de résoudre la difficile tâche de "classification de données" dans toutes les situations n'existe pas. Sa recherche est d'ailleurs très certainement utopique. Par contre, le fait de bien maîtriser le comportement d'une approche permet de l'utiliser à bon escient. Ces travaux se placent pleinement dans cet état d'esprit et vise à découvrir le potentiel d'une nouvelle approche sur des problématiques de classification de données.

Mentalement, le processus de classification peut se concevoir assez simplement avec une étape de récolte, de mesure, d'analyse d'information suivie d'une étape de synthèse et de prise de décision. L'homme est habitué à cette approche car il la pratique couramment dans sa vie de tous les jours voire même inconsciemment. Sa transposition dans un système informatique fait rapidement apparaître des difficultés, notamment liées à la recherche d'une représentation formelle de concepts principalement quantitatifs et abstraits. Dans ce contexte, les travaux présentés se concentrent principalement sur le processus d'"apprentissage" des paramètres des systèmes de classification et laissera de côté les aspects extraction des données d'apprentissage et choix du modèle mathématique de représentation.

Plus concrètement, cette thèse se propose d'étudier une approche basée sur des règles graduelles dans un système de classification de données numériques. Principalement utilisé dans le domaine du traitement de l'imprécision, ce type de règles peut également permettre de modéliser une classe. Ce type d'approche s'inscrit dans la famille des classifieurs basés sur des règles par son principe intrinsèque : les règles ont un fondement implicatif. Contrairement aux systèmes basés sur des règles conjonctives, où chaque nouvelle règle augmente le domaine "permis" pour une classe, dans le cas du système développé, chaque nouvelle règle augmente le domaine "interdit" pour la classe analysée. Les deux types de classifieurs sont donc complémentaires, les règles conjonctives apportent ce que l'on appelle de "l'information positive", alors que les règles implicatives apporte ce que l'on nomme de "l'information négative".

La manuscrit est organisé en 5 chapitres :

– **Chapitre 1 :** le premier chapitre est une introduction générale de la problématique de la classification de données. Il offre une vision d'ensemble sur ce qu'est la "classification" vue comme un processus mental. Il donne ensuite des définitions et des notions qui transposent ce

principe naturel dans un système informatique de traitement de données. Ainsi, les notions de base d'un système de classification sont introduites (ensemble de données, classe, appartenance, apprentissage, etc) et quelques schémas aident à la compréhension de la démarche générale d'un système de classification. Afin d'aider la lecture de ces notions, un exemple qui illustre toutes ces notions accompagne la présentation. Enfin, ce chapitre présentera les principales procédures d'évaluation des systèmes de classification.

- **Chapitre 2 :** le deuxième chapitre tente de présenter les méthodes de classification existantes suivant trois grandes catégories. Ce chapitre ne propose pas une énumération exhaustive des classifieurs existants, mais une vue d'ensemble des trois principes sur lesquels un système de classification peut se baser : statistique, neuronal et à base de règles. Ce choix est justifié par le fait que des classifieurs plus avancés que ceux qui sont présentés existent, et leurs améliorations sont parfois très intéressantes, mais d'un point de vue du principe, leur description apporte peu d'informations supplémentaires. La fin de ce chapitre est dédié à situer le classifieur proposé dans cette thèse au sein de sa famille d'appartenance, à savoir les classifieurs basés sur des règles.
- **Chapitre 3 :** le troisième chapitre définit le principe de base de classification qui est au cœur du système développé. L'utilisation de l'opérateur d'"implication" dans la partie "prémisse" des règles impose des contraintes importantes sur les fonctions de projection des attributs par les fonctions d'appartenance. La première partie de ce chapitre décrit le principe de l'approche, les contraintes qui doivent être respectées et son interprétation. La deuxième partie du chapitre offre une analyse détaillée de la construction des règles qui définissent le système. Ces règles sont elles aussi expliquées et interprétées.
- **Chapitre 4 :** le quatrième chapitre propose une intégration de l'approche de base dans un système complet de classification. Une première partie du chapitre est dédiée au pré–traitement des données. En effet, l'application d'un système de classification sur des données issues du monde réel est souvent sensible au bruit inhérent aux mesures. Les pré–traitements proposés sont de deux types qui peuvent être combinés si nécessaire : (1) le filtrage des points "aberrants" et (2) l'identification des ensembles de points d'apprentissage connexes au sein d'une même classe. La deuxième partie du chapitre propose deux approches d'intégration de l'algorithme dans un système complet de classification : (1) l'approche "par classes", qui encourage en cas d'incertitude le placement des points dans une classe dite d'ambiguïté et (2) l'approche "par paires d'attributs" qui va elle privilégier le rejet en cas d'incertitude. Les deux approches sont complémentaires et leur validation est étudiée sur des benchmarks. Cette première utilisation du système en situation réelle permettra également de mieux présenter la signification des notions d'ambiguïté et de rejet, précédemment introduites.
- **Chapitre 5 :** le cinquième chapitre présente trois applications réelles portant sur l'interprétation d'images numériques sur lesquelles le système a été appliqué et validé. Elles sont déclinées en deux temps : une courte introduction qui présente la problématique et le contexte de l'application, puis les résultats obtenus. Les résultats sont analysés de façon qualitative en commentant les images obtenues, mais également de façon quantitative au travers des matrices de confusion. Ces matrices de confusion auront une forme légèrement différente des matrices de confusion habituelles afin de prendre en compte les classes d'ambiguïté et de rejet générées par le système. Le comportement du système sera également analysé au travers de tracés d'histogrammes de divers ensembles de points.

Enfin, les conclusions et perspectives sur le système de classification mis en place seront développées dans la dernière partie du document. Elles seront l'occasion de résumer les avantages et inconvénients de l'approche proposée tout en en tenant compte pour se projeter vers de futurs travaux sur, par exemple, le difficile problème de l'interprétabilité des systèmes.

Chapitre 1

La problématique de la classification de données

1.1 Une activité humaine répandue

L'activité qui consiste à "classer" (ou "classifier") des objets se rencontre dans de nombreuses applications de la vie quotidienne. Cette tâche de classification n'est d'ailleurs pas toujours exprimée dans des termes scientifiques. En effet toutes les situations de la vie courante qui mènent à faire un choix passent par une catégorisation des différentes solutions envisageables sans pour autant "mathématiser" le problème. Acheter des habits, chercher son chemin, organiser sa journée sont des exemples simples, quotidiens où l'homme initie de façon intuitive un mécanisme de classification lui permettant de faire les bons choix.

Cette activité de classification passe toujours par la recherche de critères pertinents dérivant de la situation étudiée, en vue de prendre une bonne décision. Ces critères, généralement subjectifs dans le cas du raisonnement humain sont devenus numérisables au fil de l'évolution technologique. Ainsi le besoin d'outils scientifiques est apparu afin de réaliser ces classifications.

1.1.1 Bref historique

Dès le $4^{\text{ème}}$ siècle avant J.C., le problème de la classification s'est posé dans le monde scientifique. Même si la problématique était différente de celle d'aujourd'hui, un premier système scientifique de classification a été proposé par le philosophe et homme de science Aristote [26]. Son système concernait la classification des animaux, qui furent groupés par types (gr. **genera**) selon des caractéristiques similaires. Ensuite, pour chaque type, il identifia des espèces [92]. Ainsi, Aristote a identifié à cette époque deux grands types d'animaux : les animaux ayant du sang et ceux n'en ayant pas. Dans la catégorie des animaux ayant du sang, il identifie cinq espèces : quadrupèdes vivipares (comme les mammifères), oiseaux, quadrupèdes ovipares (reptiles et amphibies), poissons et baleines. La catégorie des animaux sans sang est composée, elle, de : céphalopodes (comme les poulpes), des crustacés, des insectes (en plus de ce que l'on considère "insecte" aujourd'hui, il y avait également les araignées, les scorpions, et les centipèdes), les mollusques à coquille et les "animaux-plantes", qui, dans ce système, ressemblaient d'un certain point de vue aux plantes.

Dans le cadre du système de classification proposé par Aristote, il est possible d'identifier les composants de base encore utilisés de nos jours par les systèmes de classification : les entités informationelles à classifier (dans ce cas les animaux), les attributs qui servent à faire le groupement

(la présence ou l'absence du sang par exemple) et finalement les classes de sortie (les cinq espèces d'animaux à sang, par exemple). De plus, le système d'Aristote était caractérisé par deux niveaux de traitement : une première classification qui établit les types, suivie d'une deuxième, qui identifie les espèces dans les super–classes précédemment obtenues.

Les premiers systèmes de classification ont été surtout développés dans le domaine de la biologie, où le besoin de grouper les différentes formes de vie a toujours été un des principaux centres d'intérêt des chercheurs. Parmi ceux qui ont développé de tels systèmes, les plus connus sont Carolus Linnaeus ($18^{\text{ème}}$ siècle) et surtout Charles Darwin [92] ($19^{\text{ème}}$ siècle). Pourtant, les algorithmes généraux de classification ne sont apparus qu'au cours du $20^{\text{ème}}$ siècle. On peut par exemple citer la création dans les années 1930 aux Etats Unis d'un Comité Interdépartemental de la Classification Industrielle (Interdepartmental Committee on Industrial Classification) [11]. Le principal but de ce comité fut d'établir un système cohérent et consistant de classification des différentes industries.

Depuis, des méthodes de plus en plus évoluées ont été proposées. Ces méthodes ont tendance à devenir de plus en plus générales pour être appliquées dans divers domaines. Elles sont par contre de plus en plus complexes et consommatrices de ressources.

1.1.2 Illustration à l'aide d'un exemple

Généralement, la classification peut être vue comme l'affectation d'un objet à un groupe. Le but est de séparer les entités à classifier selon des critères qui permettent d'obtenir d'une part des groupes uniformes d'un certain point de vue et d'autre part une différence claire entre deux groupes distincts.

Considérons un exemple simple pour illustrer cette problématique de classification de données. Soit le cas d'une école qui doit sélectionner ses futurs étudiants. Dans ce contexte, une commission d'admission doit associer à chaque candidat qui se présente l'une des trois classes suivantes : admis, sur liste d'attente ou non–admis. Cela revient en fait à grouper tous les candidats selon des principes de similarité dans trois catégories. Afin d'atteindre son but, la commission utilise des informations liées à l'activité antérieure des élèves (notes, comportement, activités diverses, etc), mais aussi des informations issues d'un entretien avec le candidat. Dans ce contexte, chaque membre du jury peut fournir un classement selon les points qu'il considère importants et la décision finale sera prise en tenant compte de toutes ces données.

On dispose donc d'un ensemble d'objets d'étude, qui est la totalité des candidats et d'un ensemble de groupes, qui sont les trois classes proposées : "admis", "sur liste d'attente" et "non–admis". La prise de décision, qui est le cœur du processus, correspond à l'activité de la commission d'admission. Cette dernière doit, à partir des différentes informations disponibles, faire un choix pour chaque candidat. Un problème typique peut déjà être évoqué : quelle information est pertinente et donc doit être utilisée afin de prendre une décision ? Cette information est dans ce cas liée aux performances précédentes de l'élève, au niveau scientifique relevé lors de l'entretien, etc. Ces différents aspects de l'information s'appellent "attributs". Par exemple, un attribut peut être la note obtenue par le candidat à une matière considerée importante pour son activité future, mais aussi une appréciation générale des membres du jury par rapport à son entretien. Un autre aspect du problème qui survient alors concerne les différentes représentations possibles des attributs. Pour l'application présentée, les attributs peuvent avoir des formats différents : linguistique (s'ils sont issus des projets réalisés, des avis du jury, etc), ou numérique (comme les notes obtenues pendant le parcours du candidat). Afin d'obtenir les trois classes, il faut intégrer dans le système de prise de décision toutes ces formes d'information différentes.

1.1.3 Classification – processus mental

Lorsqu'il est mis en situation de prendre une décision, l'être humain contruit d'une manière inconsciente et intuitive un système de prise de décision qui est illustré dans la figure 1.1.

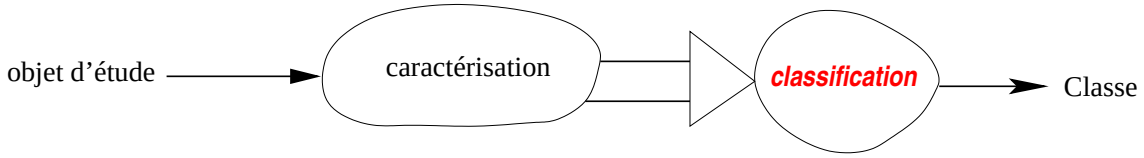


FIGURE 1.1 – Processus mental de la classification de données

Tout d'abord, le sujet humain est confronté avec l'objet d'étude, qui est en fait l'objet de la décision qui doit être prise (élèves à admettre, habits à acheter, chemin à choisir, tâches à accomplir, etc). Puis, en utilisant des **connaissances** qu'il a accumulé tout au long de sa vie, il associe à cet objet des caractéristiques qui lui permettent de le positionner par rapport à des objets "référence" qu'il connaît et qu'il est capable de situer parmi les classes de décision. Ensuite, il compare ces caractéristiques avec les caractéristiques des objets–référence qui sont conformes à ses souhaits (**objectifs qualitatifs**). Si cette comparaison est positive l'être humain prend la décision d'accepter l'objet d'étude comme conforme à ses exigences, autrement l'objet est rejeté.

Les systèmes automatiques de classification essayent d'imiter ce comportement et d'obtenir des résultats au moins aussi pertinents qu'un expert humain. Afin d'atteindre ce but, les processus actifs de la prise de décision humaine sont automatisés, comme illustré dans la figure 1.2.

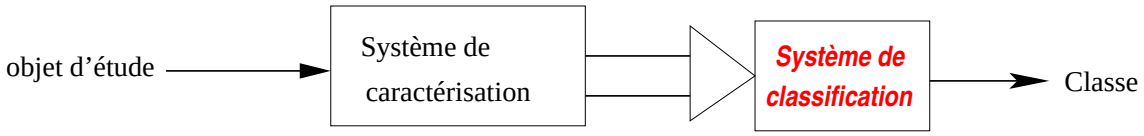


FIGURE 1.2 – Système dédié à la classification automatique de données

Le système de caractérisation des objets est l'implantation du processus d'extraction de l'information pertinente à partir des critères définis par l'utilisateur humain. La sortie de ce système est utilisée par le système de classification afin d'obtenir la décision attendue sur l'objet d'étude.

Si on se rapporte à la figure 1.1, on peut identifier les deux composantes de base d'un processus de classification, comme montré dans l'équation (1.1), où "O" représente l'ensemble des objets à étudier, "E" l'ensemble des vecteurs d'attributs d'entrée et "C" l'ensemble des classes de sortie. L'ensemble E est obtenu par un processus "$caract$" d'extraction des attributs et un processus "$classif$" de classification est ensuite utilisé afin d'obtenir la classe. On peut donc regarder la classification de données comme une application $classif : E \longrightarrow C$.

$$\begin{gathered} O \overset{caract}{\longrightarrow} E \overset{classif}{\longrightarrow} C, \; avec : \\ caract : O \longrightarrow E; \quad classif : E \longrightarrow C \end{gathered} \tag{1.1}$$

Si on considère donc un objet d'étude "obj", élément de O, il est analysé par le système de caractérisation et le résultat est un vecteur de caractéristiques "v", élément de E :

$$v = caract(obj) \tag{1.2}$$

Ce vecteur est fourni au système de classification, qui l'utilise pour prendre la décision sur la classe d'appartenance de l'objet, "c" :

$$c = classif(v) \tag{1.3}$$

Dans la suite de ce document, seule la partie "système de classification" est analysée et détaillée. La manière de choisir et représenter les attributs (le système de caractérisation) ne fait pas l'objet des travaux de recherche effectués dans le cadre de cette thèse. Il sera donc considéré que l'on dispose déjà de ces attributs, c'est–à–dire qu'un vecteur d'attributs, élément de E, est disponible pour tout objet d'étude à classer.

1.2 Objectifs et bénéfices d'une classification automatique

De nombreux domaines d'application utilisent les systèmes de classification. Afin d'offrir une vision d'ensemble sur les applications concrètes de la classification, on propose de grouper ces applications selon leurs objectifs.

On peut ainsi énumérer de façon systématique les bénéfices qui peuvent être obtenus par l'utilisation d'un système adéquat bien maîtrisé. Les objectifs principaux de la classification seront présentés de manière succincte, des détails pouvant être trouvés dans les références indiquées. Pour chaque application particulière, il y a généralement des méthodes qui sont privilégiées par les experts du domaine. En fait les algorithmes qui se trouvent à la base d'une méthode de classification sont plus ou moins adaptés à un type de données, et finalement à une application particulière.

1.2.1 Optimiser les coûts d'une expertise délicate

On peut utiliser la classification de données afin d'aider à la prise de décision dans des domaines où le coût (financier, humain, etc.) d'une décision erronée est très élevé. Une évaluation automatisée est utile, pas nécessairement pour établir la direction qui doit être poursuivie, mais pour indiquer un possible degré d'intérêt pour les différentes directions envisageables.

Une application possible de la classification pour l'aide à la décision porte sur la prospection pétrolière, où l'objectif est de préciser si les couches du sous–sol dans une région étudiée peuvent contenir des réserves de pétrole qui sont exploitables [150]. La classification se fait dans ce cas à partir d'images sismiques, qui sont des images 3D représentant le sous–sol. La méthode d'acquisition habituellement utilisée est basée sur la réflexion : on produit des ondes sonores qui se propagent et sont réfléchies par les différentes couches du sous–sol. Les réponses sont enregistrées par des géophones très sensibles et le modèle 3D du sous–sol est généré par des ordinateurs très puissants. Après l'acquisition, les informations doivent être mises sous une forme interprétable. L'extraction des caractéristiques ne se fait pas d'une manière standardisée et la synthèse d'informations s'avère être une tâche très difficile. La qualité de la classification peut être beaucoup améliorée par l'intervention des experts, qui peuvent par exemple choisir les attributs optimaux, c'est–à–dire qui offrent un degré maximal de séparabilité ou qui sont très pertinents pour une classe donnée, etc. L'application de la classification dans la prospection pétrolière est une application industrielle de grande importance financière. Les résultats de la classification peuvent indiquer les régions où l'on peut commencer la prospection pétrolière. Les coûts d'une telle action sont importants, ce qui confère à la classification un caractère très sensible.

L'application de la classification de données dans le domaine médical est aussi un exemple d'aide à la décision. Elle concerne principalement le diagnostic de différentes maladies, malformations, lésions etc. Si dans la plupart des applications médicales l'objectif de la classification est similaire (évaluation du risque d'une éventuelle pathologie), les données et informations prises en compte dans l'évaluation sont de nature très variable (images, signaux, dosages, symptomes exprimés, etc). Le choix d'un classifieur doit donc être adapté aux exigences du domaine étudié. L'interaction du système de classification avec les étapes d'acquisition et de représentation des données (taille de l'ensemble de données, nombre d'attributs considérés, nombre de classes à distinguer) doit aussi être prise en compte.

Une application particulièrement répandue est la classification de tumeurs [108, 3]. En géneral dans ce cas entre 25 et 31 attributs issus de mesures spécifiques sont tout d'abord extraits des données d'origine. Une étape antérieure à la classification, de pré–traitement, peut être ensuite réalisée afin de réduire la dimensionalité de l'espace de travail. Un autre type d'application dans le domaine médical est l'identification des fractures ou de l'ostéoporose dans des radiographies osseuses [31]. L'ostéoporose est une dégradation des os qui augmente le risque d'une future fracture. Il est prouvé qu'un médecin peut manquer des cas critiques en inspectant rapidement des radiographies [103]. En effet, si la majorité des radiographies qu'il analyse ne présente ni fractures ni ostéoporose, il est possible qu'il ne remarque pas les rares radiographies qui présentent un risque. Dans ce contexte, il devient intéressant d'analyser automatiquement les radiographies pour isoler les cas à risque potentiel et les présenter au médecin [152].

Une autre application, moins répandue, est la génétique et la biologie moléculaire [106, 145], où le problème principal est constitué par le manque de standardisation des données. En plus, pour la plupart des systèmes de classification, l'entité de base à classifier (soit la chaîne d'ADN, soit le gène) doit être mise sous une forme numérique afin de faciliter le traitement. L'étape d'extraction des caractéristiques est donc très importante. Il est aussi nécesssaire de faire une sélection pertinente des entités à traiter afin de réduire la taille des données [101, 118].

Les indicateurs qui sont les plus utilisés dans le domaine médical sont liés aux pourcentages des cas positifs (malins) identifiés ou non–identifiés. Il est plus important de connaître ces pourcentages que le taux de classification correcte, parce que le coût d'une décision fausse est différent selon qu'elle concerne un cas positif ou négatif : pour un cas bénin qui est classifié comme malin, des tests supplémentaires sont nécessaires et il y a de fortes chances que l'erreur soit découverte, mais si un cas malin est classifié comme bénin, les tests supplémentaires ne sont pas effectués, ce qui nuit à la santé du patient [160]. Malheureusement, assez souvent le rapport entre le nombre d'exemples dans les deux classes (benin/malin) est très grand et la classe comportant le plus d'exemples d'apprentissage est plus facile à "apprendre".

1.2.2 Améliorer la production et la productivité

La classification de données peut aussi être appliquée dans différents domaines de l'industrie comme le contrôle de la qualité, en particulier pour la détection préventive de défauts. Le but de la classification est dans cette situation d'identifier les défauts ou les pièces sur le point de se dérégler, ce qui n'est pas toujours évident. La détection et la prédiction des défauts dans les systèmes industriels avant leur plein avancement peut permettre la programmation des réparations au moment le plus approprié, la commande des pièces à remplacer avant leur panne et ainsi la réduction des arrêts inattendus dans la production [110]. Un autre avantage important est la possibilité d'éviter les accidents de travail. Pour une application dans le domaine de l'agriculture, il est possible d'analyser les animaux et/ou les plantes et de maximiser les bénéfices en choisissant les éléments qui sont les

plus performants selon un critère.

Selon le problème analysé, la classification peut se baser sur différents types de données : images radiographiques, paramètres donnés par des capteurs, images spectrales, données statistiques. L'analyse automatique de l'information peut arriver à déterminer d'une manière automatique si des pièces industrielles, des milieux industriels ou agricoles, des cultures ou des populations agricoles sont en conformité avec un ensemble de spécifications de qualité et de caractéristiques imposées.

Comme exemples d'application on peut citer la détection automatisée de défauts en moulages et soudures d'aluminium [67], la gestion de troupeaux de vaches laitières par l'analyse des courbes de lactation [122], la gestion d'élevage de porcs par le choix des bêtes à sacrifier [86], les systèmes de diagnostic pour les machines tournantes [110], la détection d'incendies [96], la détection des maladies dans les récoltes [116], ainsi que d'autres classifications dans le domaine alimentaire [134].

1.2.3 Sécuriser les hommes et les biens

La classification de données intervient également dans le domaine de la sécurité. On peut citer d'abord les applications militaires, les accès sécurisés à l'information gouvernementale et à l'information confidentielle des grandes compagnies. Malheureusement, on n'a pas beaucoup d'information sur les solutions adoptées, parce que toutes les organisations qui s'occupent de ce type d'applications essaient de garder le secret sur leurs méthodes et outils. Généralement, trois niveaux de sécurité existent : confidentiel, secret et secret strict. C'est au classifieur de décider le niveau de sécurité qui s'applique à chaque document ou n'importe quelle autre forme d'information qui existe dans la base de données.

Les applications qui peuvent être trouvées dans la littérature disponible publiquement traitent principalement de la reconnaissance des visages, des signatures, de l'iris et de la voix. Ce type de classification peut aider à automatiquement identifier les imposteurs (par exemple dans les aéroports, les gares, etc.), mais aussi à donner des droits d'accès personnalisés (accès dans une institution, accès aux données confidentielles, autorisation d'opérations bancaires, etc).

La reconnaissance de la signature est utilisée dans les applications de sécurité actuelles (les opérations bancaires, les documents officiels, etc.), mais aussi dans l'authentification des auteurs des œuvres artistiques (les signatures des peintres, l'authentification des manuscrits) [45]. Afin de réaliser la classification il faut d'abord extraire les caractéristiques numériques qui doivent ensuite être traitées [112, 9]. Les problèmes typiques pour la classification des signatures sont liés principalement à l'inexactitude de l'auteur lui–même. L'individu peut avoir des signatures très différentes dans des conditions différentes : température, état de fatigue ou de stress, changement de l'objet d'écriture. De plus, la signature et le modèle d'écriture changent de manière naturelle et continue dans le temps. Des exemples les plus récents possibles sont donc requis pour optimiser la reconnaissance.

Une autre application est la reconnaissance de visages [121]. Des mesures des différentes proportions entre des points significatifs (les yeux, le nez et les lèvres) sont généralement recherchées. Plus récemment des éléments plus complexes ont été utilisés, comme des informations liées à la texture et à la forme. Ces informations permettent d'aboutir à des classifications plus complexes, comme par exemple l'utilisation de l'expression du visage afin de déduire l'état d'esprit de l'individu, identifier le sexe de l'individu, etc [115]. Les nouvelles méthodes peuvent aussi être appliquées dans d'autres domaines que la reconnaissance des visages. Une direction d'étude actuelle est l'analyse des images 2D obtenues sous différents angles afin de reconstruire les visages dans des modèles 3D [15]. Les buts principaux de la classification des visages sont énumérés dans [144] : le contrôle de documents, le contrôle d'accès et l'utilisation d'une photo occasionnelle afin d'obtenir des informations diverses

(nom, occupation, etc) à partir d'une base de données.

Une alternative possible dans le problème d'identification des personnes est la reconnaissance d'iris. Bien que l'iris ait de très petites dimensions (diamètre d'environ 11mm) et que l'obtention d'une image de celui-ci d'une qualité satisfaisante puisse être difficile (la distance entre l'appareil-photo et l'individu doit être inférieure au mètre), l'iris présente les avantages d'être différent pour chaque personne, d'être très bien protégé par l'environnement et d'être stable dans le temps [156]. L'image de l'iris n'est pas très sensible à l'illumination et à l'angle de visualisation (la déformation de texture provoquée par la dilatation des pupilles peut être traitée [34]). La classification basée sur l'iris est principalement un problème de classification de textures.

Dans le même domaine, on peut aussi placer l'identification de la voix et l'analyse des empreintes digitales. Pour l'identification de la voix, il faut préciser que même si les résultats sont très bons pour les données d'apprentissage, l'application des systèmes dans un contexte réel est difficile, car il est nécessaire de tenir compte de plusieurs paramètres : l'environnement d'acquisition, l'état de santé de l'orateur, etc. La reconnaissance des empreintes digitales est une méthode très stable, parce que la structure d'empreintes digitales est très bien individualisée pour chaque être humain et qu'elle se conserve dans le temps [109].

Dans le domaine de la sécurité des hommes et des biens on peut aussi inclure le traitement et la classification des images satellitaires radar. L'imagerie radar peut être utilisée pour surveiller l'état d'un territoire choisi : la cartographie des changements dus à la pollution ou à la dégradation environnementale, la cartographie de la végétation et la surveillance des zones urbaines, industrielles ou agricoles. Les résultats de la classification des images radar sont utilisés afin de réaliser des cartes actualisées et précises pour des régions difficiles à tracer de manière classique, comme des forêts tropicales, des secteurs brûlés ou des zones de hautes montagnes [35].

Le domaine multimédia est un autre domaine d'application. La classification des messages électroniques sur Internet est faite afin d'identifier et filtrer les courriers indésirables (Spams) [24]. Mis à part la gêne des utilisateurs, les messages indésirables peuvent facilement provoquer la panne des serveurs de messagerie. La quantité de messages de ce type peut atteindre 30% du trafic total. Les performances du réseau chutent donc avec l'augmentation de ce trafic. De plus, ces messages occupent inutilement l'espace mémoire des serveurs. Les critères d'évaluation des algorithmes appliqués dans ce domaine concernent non seulement la précision de la classification, mais aussi la vitesse d'exécution, parce que le système doit réagir en temps réel.

1.2.4 Explorer, rechercher et découvrir des informations

La découverte et l'exploration d'informations concerne principalement l'analyse de bases de données de très grande taille, surtout pour les fichiers ayant un contenu divers, avec des composantes audio, vidéo et/ou linguistique. Il peut donc être nécessaire de faire une analyse complexe sur les trois composantes : audio, vidéo et linguistique. La quantité d'informations disponibles sur l'Internet par exemple est très importante, mais à cause du manque de standardisation dans le domaine, les documents utiles ne sont pas toujours trouvés par les outils automatiques. La classification est donc utile dans ce cas afin de trouver les ressources du web qui correspondent à certains critères. Ainsi les utilisateurs peuvent localiser des informations recherchées d'une manière structurée et hiérarchisée.

Pour arriver à classifier les fichiers audio disponibles sur l'Internet, on dispose habituellement d'informations supplémentaires ("metadata") qui aident dans le processus de recherche. L'existence des fichiers qui ne disposent pas de ces informations peut rendre le processus plus difficile et plus long. Dans ce cas, la classification est basée seulement sur l'information audio contenue dans le fichier. La

classification est faite à partir de quelques paramètres, comme par exemple le tempo, des paramètres du signal de bas-niveau (taux de passage par zéro, bande du signal, centre spectral, énergie du signal, etc), des coefficients mel–cepstraux [59], des caractéristiques obtenues à partir d'un modèle perceptif, etc. Ces paramètres peuvent être utilisés afin de classifier des fichiers contenant de la musique ainsi que des fichiers acoustiques de toute sorte. L'approche basée sur le tempo est utilisée généralement pour la classification des pièces musicales de danse, mais les autres caractéristiques peuvent également être utilisées pour des problèmes plus généraux. Il est important de préciser que pour ce type d'application de la classification, l'extraction des paramètres est, là encore, une étape importante qui consiste à trouver les caractéristiques pertinentes.

Dans le cadre de la classification des images et des séquences video, des difficultés importantes peuvent apparaître à cause du bruit, des petites rotations de l'image, des variations d'illumination. Lors de la conception de ces systèmes de classification il faut tenir compte de tous ces problèmes. Une partie importante des problèmes de classification des images consiste à faire une recherche sémantique. Dans ce cas, le but de la classification est de retrouver des images qui contiennent des objets d'une certaine forme ou/et couleur, qui satisfont un critère de distribution des niveaux de gris ou des couleurs, qui sont prises à l'intérieur ou à l'extérieur (pour les images d'extérieur on peut avoir par exemple des paysages de mer, donc une grande proportion de pixels dans une nuance de bleu ; des paysages de campagne, donc beaucoup de pixels dans une nuance de vert), qui sont semblables à une image–cible d'entrée, etc. Pour ces applications, l'interface avec l'utilisateur est très importante afin de bien définir les paramètres de recherche. Une étape importante dans la classification vidéo est la définition des paramètres qui caractérisent de façon pertinente une séquence [64, 97].

1.3 Formalisation et vocabulaire spécifique aux systèmes de classification

Pour travailler sur les systèmes de classification, un ensemble de termes et notations est nécessaire pour exprimer les différentes étapes et les différentes informations manipulées. Cette partie se propose d'en faire une présentation. Quelques définitions de la littérature, ainsi que des illustrations s'appuyant sur l'exemple évoqué au paragraphe 1.1.2 (recrutement pour l'entrée dans une école supérieure) seront présentées afin de mettre en évidence l'architecture générale d'un système de classification.

1.3.1 Définitions existantes

Différentes définitions de la classification de données sont proposées dans la littérature. Certaines d'entre elles sont dédiées à un domaine d'application particulier, d'autres sont plus générales mais associées à une activité donnée. Par exemple, la notion de classification est souvent considérée comme similaire à celle de "prise de décision". Dans ce contexte, Ruta et al. [136] l'associe à l'opération qui a pour but final de produire des décisions pertinentes à partir d'une quantité minimale d'informations/données d'entrée.

Un autre terme souvent attaché à la notion de classification est la "taxonomie". Ainsi, pour Rich [131] une classification organisationelle fournit la base d'une recherche efficace, par séparation de l'univers continu des organisations en catégories discrètes et collectives adaptées à une analyse détaillée. Dans ce contexte, la classification offre la possibilité de reconnaître et/ou découvrir des structures fondamentales et des relations entre ces structures.

Une définition assez complète et spécifique est donnée par Fumera et al. [51] : un classifieur pour N classes a pour but de diviser l'espace des attributs en N régions de décision $D_i, i = \overline{1, N}$ de telle manière que les points de la classe ω_i appartiennent à la région D_i. Ces régions sont définies de façon à maximiser la probabilité d'identification correcte, habituellement nommée "l'exactitude" du classifieur. Cette définition est exprimée d'une manière beaucoup plus formelle que les précédentes et elle introduit un vocabulaire spécifique aux systèmes de classification. Tout d'abord la notion de classification implique tout de suite la notion de "classe". Une classe peut être définie comme une subdivision de l'univers de discours qui est composée par des individus similaires selon un ou plusieurs critères. Un autre terme spécifique est le terme d'"espace des attributs". Les attributs sont des formes de représentation des critères qui servent à identifier des individus similaires. Selon l'approche choisie, ils peuvent être combinés pour résoudre le problème donné.

La notion de classification est également exprimée en fonction du domaine d'application comme par exemple l'analyse des images numériques. Dans ce contexte, I. Bloch [12] précise que la classification des images est le processus consistant à partitionner une image digitale en plusieurs régions (ensembles de pixels). Le processus de classification est alors bien distingué de la "segmentation" qui, elle, représente le processus de simplification et/ou de changement de la représentation d'une image pour la rendre plus simple à analyser. Ce processus est typiquement utilisé afin d'identifier des objets ou des frontières entre les objets. Les pixels d'une même région respectent un critère de similarité, qui peut être lié à la couleur, l'intensité, la texture, etc. En même temps, les pixels des régions adjacentes sont fortement différents selon ces mêmes critères. La classification de l'image est une segmentation suivie par une reconnaissance des régions délimitées. Cette étape supplémentaire va apporter une certaine signification aux régions. Cela va généralement faciliter l'interprétation par un utilisateur (expert) humain.

Un domaine particulier très proche de la classification de données est celui de la reconnaissance des formes (pattern recognition) [44]. La notion de forme est définie par Jain et al. [81] comme étant opposée au chaos, c'est–à–dire une entité au moins vaguement définie et à qui l'on peut associer un nom. Le but de la reconnaissance des formes est alors d'identifier une forme d'entrée comme un membre d'une classe [154]. Les applications liées à la reconnaissance des formes sont très nombreuses et elles couvrent de très larges domaines applicatifs.

Le principe général (processus mental) de la classification de données, illustré sur la figure 1.1, peut être synthétisé comme l'affectation d'une classe à un objet en fonction des caractéristiques de cet objet. Plusieurs objets appartiendront à la même classe si leurs caractéristiques sont similaires et à des classes différentes dans le cas contraire. Les difficultés qui apparaissent au cours de cette démarche portent sur la définition et l'évaluation de la notion de similarité mais également sur la découverte de la relation liant les objets aux classes recherchées. La caractérisation des objets d'étude est également une étape importante liée principalement aux mesures et pour laquelle des traitements spécifiques sur les données peuvent être nécessaires. Dans le domaine strict de la classification, cette étape est généralement considérée comme déjà effectuée. Bien évidemment, la qualité globale du système dépendra de la pertinence et du pouvoir discriminant des informations issues de cette étape.

1.3.2 Architecture et principe général

On distingue généralement deux stades dans le cycle de vie des systèmes de classification qui ont été précédemment illustrés sur la figure 1.2 : un stade d'apprentissage et un stade d'utilisation.

1.3.2.1 Architecture et principe général de l'apprentissage

L'étape d'apprentissage imite le comportement humain : à partir d'un ensemble d'apprentissage A le système construit un modèle de représentation de la connaissance. En fonction du choix conceptuel que l'on fait, ce modèle peut se présenter sous différentes formes : règles de classification, réseaux neuronaux, arbres génétiques, hypersurfaces SVM, etc. Ce choix conceptuel est essentiel pour la suite de l'apprentissage. Il doit être fait de façon à assurer une cohérence entre l'objectif visé et les moyens mis en œuvre pour l'atteindre. Ainsi, une conception optimale du système de classification nécessite des connaissances a priori sur les caractéristiques de l'application pour laquelle le système est conçu, ainsi que sur les propriétés, avantages et inconvénients des différents modèles de représentation possibles. Une phase d'identification de la difficulté de la tâche de classification et des problèmes typiques susceptibles d'apparaître [148] peut notamment guider le choix d'un type de modèle de représentation.

L'ensemble d'apprentissage A est l'élément de base dans le processus d'apprentissage. Si on se rapporte à la représentation analytique de l'equation (1.1) on peut identifier deux cas possibles :

- $A \subset E$: le système dispose des caractéristiques (vecteurs d'attributs) des exemples, mais il n'a aucune connaissance a priori sur la classe d'appartenance des données qu'il reçoit. Dans ce cas–là, le système doit construire lui même une répartition de ces données dans des classes individualisées, séparables et significatives. Cette approche peut être trouvée dans la littérature sous le nom d'"**apprentissage non–supervisé**" ou bien de "clustering" [10, 82].
- $A \subset E \times C$: chaque élément de l'ensemble d'apprentissage est donné sous la forme d'une paire composée d'un vecteur d'attributs, souvent appellé "point d'apprentissage", et de la classe de sortie attendue. Dans ce cas–là, le système doit construire le modèle de façon à affecter un nombre maximal de points d'apprentissage à leur classe de sortie. Cette approche est connue sous le nom d'"**apprentissage supervisé**" [38].

La nature de l'ensemble d'apprentissage disponible détermine donc le type d'apprentissage (supervisé ou non supervisé) à réaliser. Reste alors à choisir une méthode d'apprentissage adaptée au modèle de représentation choisi et aux données d'apprentissage disponibles. Il faut aussi mentionner que selon l'application et les performances requises, ces deux choix peuvent être remis en cause afin d'améliorer au maximum les résultats obtenus.

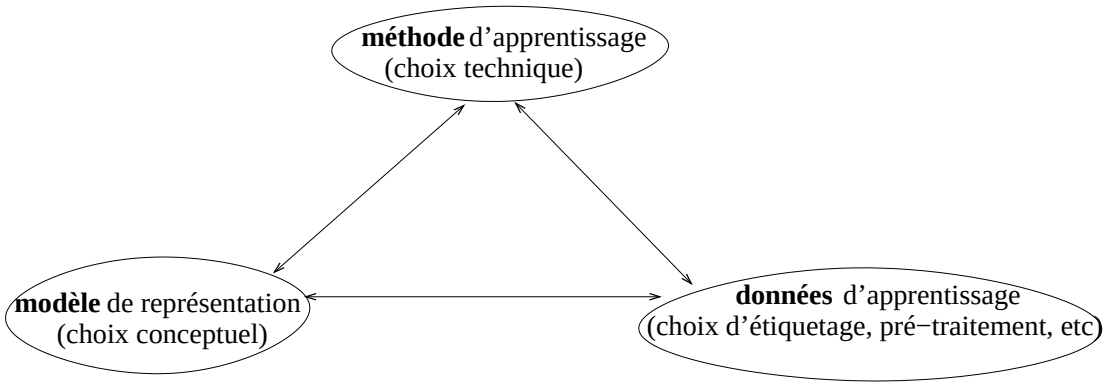


FIGURE 1.3 – Le contexte d'un système d'apprentissage

En résumé, dans le cadre de la classification, un système d'apprentissage est caractérisé par un triplet (modèle, données, méthode), comme illustré dans la figure 1.3. Il est généralement associé à des choix conceptuel et technique interdépendants et fortement liés à la nature des données disponibles. Il est également possible d'agir sur l'acquisition même des données d'apprentissage. Plus classiquement,

on se limitera à une étape de pré–traitement de ces dernières.

La figure 1.4 présente l'approche généralement utilisée par les systèmes d'apprentissage, chargés de réaliser l'apprentissage des systèmes de classification. Au cœur de ces systèmes, apparaît le processus d'apprentissage, qui recherche dynamiquement les paramètres optimaux du système de classification, c'est–à–dire ceux qui conduisent à une séparation optimale des différentes classes analysées.

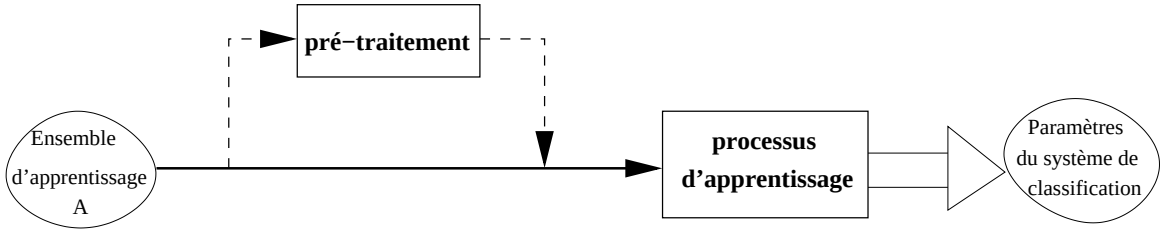


FIGURE 1.4 – Principe général des systèmes d'apprentissage

Sur la figure 1.4, il apparait également un composant supplémentaire de pré–traitement [33] de l'ensemble d'apprentissage. Ce dernier n'est pas obligatoire et son utilité dépend essentiellement de la qualité de l'ensemble d'apprentissage A. Le pré–traitement peut d'ailleurs être directement intégré dans l'étape d'extraction de l'information, mais parfois le système d'apprentissage lui–même doit traiter l'ensemble de données qui lui est fourni de façon à eventuellement inclure l'information liée à ce traitement dans les paramètres du système de classification pour la phase d'utilisation.

L'identification des attributs significatifs et discriminants pour le problème donné peut être une question assez difficile si on ne dispose pas de connaissances antérieures. Un exemple qui sert à la classification peut être caractérisé par plusieurs attributs parmi lesquels seuls certains peuvent être discriminants et significatifs pour l'application, voire pour uniquement certains composants du système de classification.

On peut voir le pré–traitement des données selon deux points de vue : un pré–traitement axé sur les attributs (ensemble E) ou sur les points (vecteurs) d'apprentissage (ensemble A). Le but du pré–traitement des attributs est souvent de réduire la dimension de l'espace d'entrée pour faciliter les traitements ultérieurs. En effet, l'analyse et la classification des ensembles de données multidimensionnelles peuvent devenir très longues et consommatrices de mémoire, surtout pour les systèmes qui prennent en considération des mesures relationelles entre différentes combinaisons d'attributs. De plus, il n'est pas garanti que les résultats obtenus sur les données multidimensionnelles soient meilleurs que ceux correspondant au même ensemble modifié par une étape antérieure d'extraction d'attributs, même si une partie de l'information disponible sur l'ensemble original est perdue pendant le processus. Le pré–traitement des attributs peut donc avoir deux aspects : la **sélection** des attributs (quand une partie des attributs, considerés non–pertinents, est complétement ignorée par le système) et l'**extraction de l'information pertinente** à partir des attributs disponibles (quand de nouveaux espaces des attributs, de dimension réduite, sont calculés à partir des attributs disponibles).

La sélection des attributs consiste à choisir parmi la totalité des attributs qui sont disponibles dans la base de données ceux qui ont une signification pour l'application et qui conservent l'essentiel de l'information présente dans l'ensemble de données initial. Ainsi, on peut obtenir des résultats optimaux en réduisant le temps de calcul et les ressources utilisées.

L'autre aspect du pré–traitement du point de vue des attributs est l'extraction de l'information, qui consiste à analyser la totalité des attributs disponibles afin d'éliminer la corrélation entre les attributs, d'extraire seulement l'information pertinente et la présenter sous une forme compacte et adaptée au système d'apprentissage. Il s'agit donc d'identifier des liens de dépendance entre différents

attributs et/ou valeurs de ces attributs. Les principales méthodes d'extraction de caractéristiques généralement utilisées sont la transformée de Karhunen-Loëve [58] (l'analyse en composantes principales, une méthode très connue dans le contexte de la décorrélation des signaux), la transformée de Fourier discrète [14], la transformée cosinus discret [141] (ou sinus discret), la transformée en ondelettes [104] ou bien la matrice de cooccurrence ou la matrice "run-length" [125].

Ces deux approches du pré–traitement sur les attributs ont des avantages et des inconvénients. La deuxième est très efficace d'un point de vue mémoire nécessaire pour traiter l'ensemble de données, mais elle a aussi le grand désavantage d'éliminer la signification des attributs. En l'appliquant, on change le système de représentation des attributs et donc un expert du domaine de l'application ne peut plus interpréter les résultats de la même manière que s'il se situe dans un cadre de travail familier où il peut associer les axes des attributs avec des notions spécifiques à son domaine, qu'il sait aborder et comprendre. Généralement, l'étape d'extraction de l'information pertinente n'est pas réalisée dans ce type d'application où l'expert exige d'avoir la possibilité d'interpréter et analyser les données d'apprentissage, même si d'un point de vue ressources et temps de calculs elle pourrait beaucoup augmenter les performances du système. L'utilisabilité du système par les experts du domaine reste primordial. Par contre, si les attributs de l'application sont, par définition, abstraits et n'ont donc pas une signification interprétable, une approche par extraction plutôt que par sélection est souvent préférable.

La deuxième vision du pré–traitement des données, c'est–à–dire le pré–traitement des points d'apprentissage consiste le plus souvent à réaliser une épuration en éliminant les points de l'ensemble d'apprentissage considerés comme "aberrants" selon des critères bien définis.

1.3.2.2 Architecture et principe général de l'utilisation des systèmes de classification

Une fois les paramètres du système obtenu, la classification de nouvelles données peut être réalisée. La phase d'utilisation du système de classification consiste à réaliser la classification de nouvelles données d'entrée, jusqu'à maintenant inconnues pour le système, en utilisant la méthode de classification choisie avec les paramètres appris dans l'étape précédente, comme montré sur la figure 1.5. Le processus de classification est alors composé de deux étapes différentes : une d'évaluation de l'adéquation du point analysé aux classes apprises et une de prise de décision. La sortie de la première étape peut être à ce niveau une classe unique, correspondant en fait au cas où un seul coefficient d'adéquation est non nul, mais plus généralement elle sera un vecteur contenant des degrés (probabilités) d'appartenance aux classes analysées. Typiquement, on peut considérer cette sortie comme un vecteur avec des valeurs $\in [0, 1]$, dont la dimension est donnée par le nombre de classes considérées.

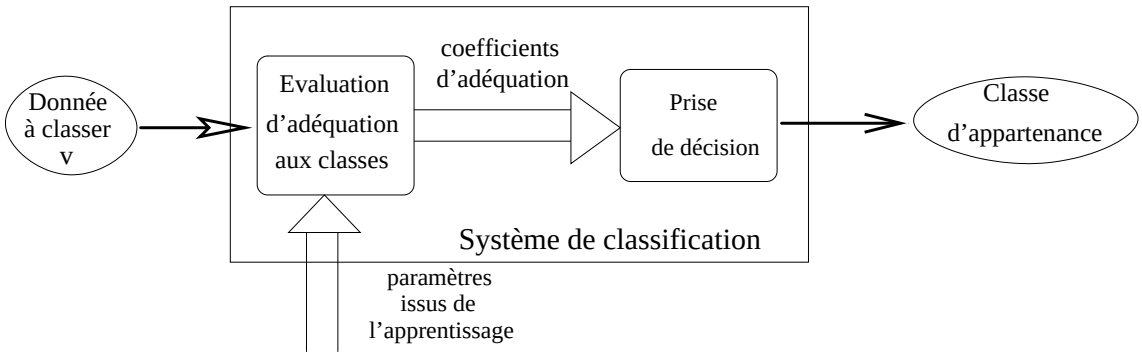


FIGURE 1.5 – Principe général d'utilisation des systèmes de classification

La prise de décision traite ce vecteur de coefficients d'adéquation afin d'arriver à une décision

finale, qui est l'affectation de l'entrée à une et une seule classe de sortie. Pour cette étape, les sorties du système d'analyse d'adéquation, toutes relatives à la même entrée, sont généralement combinées afin d'augmenter la fiabilité du résultat final.

1.3.2.3 Retour sur l'admission des candidats

Les notions et principes des deux étapes des systèmes de classification, apprentissage et utilisation, peuvent être illustrés sur l'exemple de sélection d'étudiants à l'entrée d'une école supérieure, présenté dans le paragraphe 1.1.2.

Dans cet exemple, les données d'apprentissage sont extraites des dossiers des étudiants qui se présentent au concours d'admission. Les vecteurs d'attributs regroupent donc des données de nature différente, numérique et linguistique, et sont évidemment non–étiquetés. On se retrouve alors face à un problème d'apprentissage non–supervisé.

On peut imaginer aussi le cas d'un apprentissage "dynamique", où on utilise l'expérience acquise pendant les années précédentes : une base de données d'apprentissage peut être établie à partir des dossiers acceptés les années précédentes, en prenant également en compte les résultats obtenus par les étudiants acceptés et ayant effectivement intégré l'école. Dans ce cas–là, les données d'apprentissage sont étiquetées et le problème se situe dans le cadre d'un apprentissage supervisé.

Un modèle de représentation basé sur des règles peut être choisi. Ces règles ont une forme linguistique pour les données linguistiques et une forme basée sur des seuils pour les données numériques (moyennes minimales nécessaires pour être admis). Le processus d'apprentissage consiste à obtenir ces règles en utilisant soit les informations issues des années précédentes soit une hiérarchie des dossiers de l'année en cours, ou encore une combinaison des deux.

L'épuration de l'ensemble de données consisterait à éliminer dès le début les dossiers qui sont complètement incompatibles avec le profil requis par l'école. Par exemple, une école qui forme des ingénieurs spécialisés peut ne pas accepter le dossier d'un candidat qui a de très faibles résultats dans les matières de base du domaine, comme les mathématiques et la physique, ni celui d'un candidat qui a eu un parcours complétement axé sur les sciences humaines.

Plusieurs informations sont généralement disponibles dans les dossiers fournis par les candidats. Parmi ces attributs on peut citer ceux liés à la nationalité, le sexe, l'âge, les projets antérieurs qui ne sont pas forcément liés à l'activité que l'étudiant va poursuivre en cas d'admission à l'université où il dépose sa candidature. Ce type d'information n'a pas une valeur réelle pour le problème posé et la commission va donc intentionnellement l'ignorer. Sans le savoir, en ignorant une partie de l'information qui lui est fournie, le jury fait ainsi une sélection des attributs pertinents.

Du point de vue de l'extraction de l'information, on peut identifier dans l'exemple proposé les avis formulés par les différents membres du jury de sélection. Chaque membre est intéressé par des aspects différents de la formation du candidat : un professeur de mathématique va fournir une évaluation des capacités de structuration et formalisation du candidat, un professeur de physique va plutôt évaluer la capacité à appréhender des phénomènes physiques et ainsi de suite. Ces évaluations sont formulées dans un language naturel, mais elle peuvent être réunies dans un indice qui donnera le degré d'appréciation générale du candidat. L'indice le plus élevé correspondra à l'élève qui est le plus proche des exigences de tous les membres du jury. Ce que l'on peut remarquer dans cette situation est la perte de signification de l'attribut extrait. L'attribut introduit qui fait la combinaison de tous les avis n'a pas une signification sémantique pour les membres de la commission, qui ne connaissent et ne peuvent interpréter que les attributs de base, exprimés en language naturel.

En ce qui concerne l'utilisation des règles d'admission ainsi mises en place, chaque donnée à

classifier est composée d'un nombre de caractéristiques de chaque candidat : les notes antérieures, les rapports de projets auxquels il a participé, les avis des membres du jury, etc. A partir de ces informations, l'adéquation de chaque étudiant est calculée et la décision peut ensuite être prise en ordonnant les adéquations de tous les candidats. La prise de décision peut être dans ce cas soit l'admission des premiers candidats, en choisissant un nombre fixe de candidats, soit l'admission des étudiants qui ont l'adéquation minimale requise.

L'approche décrite ici pour l'apprentissage et l'utilisation (y compris la prise de décision) du système de classification dédié à l'admission a ses propres caractéristiques qui peuvent la rendre plus adaptée à certains types de problèmes qu'à d'autres.

1.3.3 Fusion d'informations

La problématique de la classification de données telle qu'elle a été exprimée ci–dessus rentre pleinement dans le cadre plus général de la fusion d'informations.

Terme générique comportant de nombreuses facettes, la fusion d'informations permet la synthèse de données de toutes sortes afin d'obtenir une information sémantiquement plus riche. La figure 1.6, extraite de [126], propose un synoptique des systèmes de fusion d'informations.

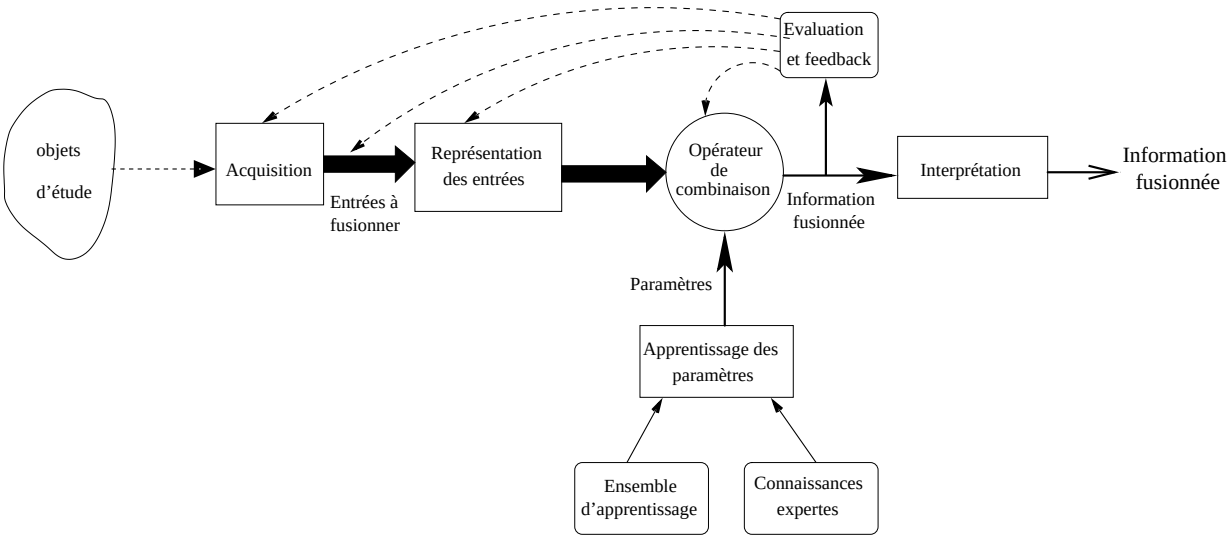


FIGURE 1.6 – Synoptique des systèmes de fusion d'informations

Sur ce synoptique, on retrouve les principales étapes qui sont également mises en œuvre dans les systèmes de classification. L'aquisition d'information et sa représentation correspondent à la partie "caractérisation" des classifieurs. L'opérateur de combinaison et l'interprétation correspondent aux systèmes de classification. Enfin on retrouve également l'apprentissage des paramètres de l'outil mathématique utilisé.

Dans le domaine de la fusion, on retrouve les grandes problématiques précédemment évoquées, comme le choix de la méthode de combinaison, la forme des données d'entrée (nature, espace de représentation, etc), le mécanisme d'apprentissage, la problématique de l'évaluation, etc.

La figure 1.7 repositionne les grandes étapes des systèmes de classification sur le synoptique des systèmes de fusion.

Dans ce contexte, la particularité d'un classifieur va porter sur le niveau des informations qu'il

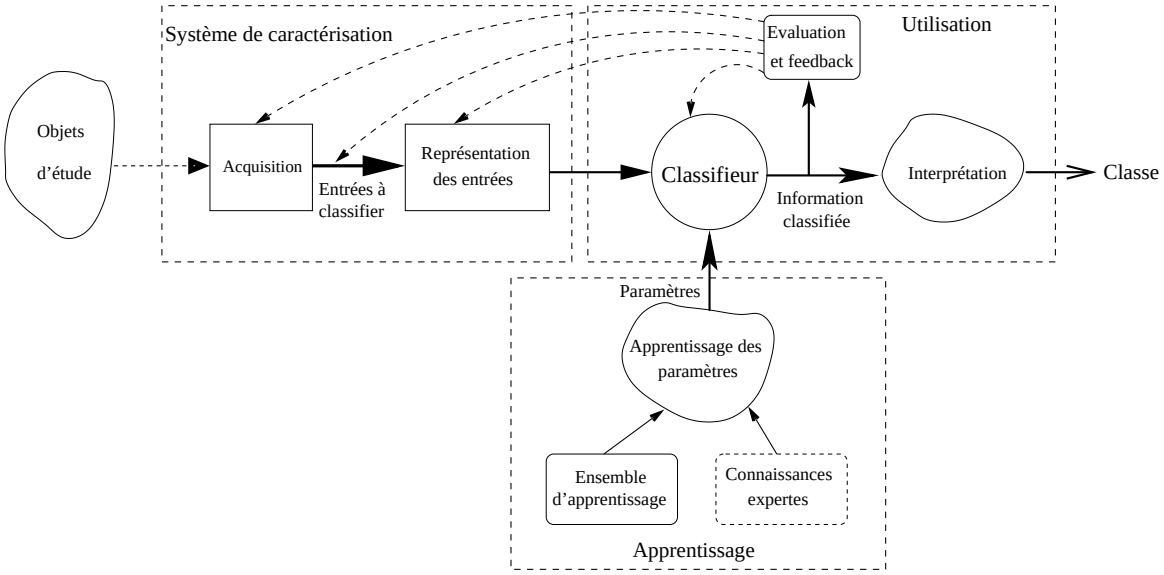


FIGURE 1.7 – Systèmes de classification et fusion d'informations

manipule. Dasarathy [32] definit trois niveaux ordonnés d'informations manipulées par les systèmes de fusion, qui sont :

- le niveau signal : consiste à travailler directement sur des données brutes
- le niveau attribut : l'information porte sur une caractéristique de l'objet étudié (généralement un pré–traitement sur les données brutes)
- le niveau décision : niveau le plus haut, il correspond à une prise de décision sur l'objet étudié (qualité, choix d'une action, etc)

A partir de ces trois niveaux, Dasarathy présente alors les systèmes de fusion en fonction du niveau des informations d'entrée et de sortie, en imposant que le niveau de sortie soit au moins égal au niveau des entrées. Ainsi, un système de classification est un système de fusion pour lequel les informations d'entrée appartiennent au niveau attribut et qui fournit une information de sortie de type décision.

1.4 Evaluation des performances d'un système de classification

1.4.1 La problématique de l'évaluation

Deux cas sont à distinguer dans l'évaluation des systèmes de classification : le premier concerne les systèmes pour lesquels on ne dispose pas de données de référence. Dans ce cas, l'évaluation est rendue difficile car la comparaison des résultats du classifieur à des références sur des points précis et bien connus n'est pas possible. Les approches généralement mises en œuvre s'appuient sur l'analyse de statistiques qui donnent par exemple le degré de séparation des classes obtenues [87] ou visent à déterminer le pouvoir organisationnel d'un classifieur à partir de données non–étiquetées [100]. Cet aspect de l'évaluation ne sera pas développé dans cette partie.

Le deuxième cas concerne les systèmes pour lesquels des données "de référence" sont disponibles. Les données de référence peuvent être soit celles qui ont servi à l'apprentissage soit d'autres

données en lien ou non avec l'application–cible du système. Dans ce contexte basé sur des données de référence, l'utilisation de benchmarks est très répandue. Les benchmarks sont des ensembles de données étiquetées en libre accès pour la communauté. Elles permettent une comparaison pertinente des performances des systèmes de classification basés sur des concepts ou des méthodes différents.

Par exemple dans un contexte de classification supervisée, on peut considérer le cas des données "Iris" disponible sur le site "http ://archive.ics.uci.edu/ml/datasets.html", où chaque exemple est déjà associé à une des trois classes pré–définies. Chaque exemple est caractérisé par 4 attributs qui peuvent servir pour faire la distinction entre 3 classes de fleurs. Ces attributs sont la longueur et la largeur des sépales et des pétales. Pour comparer les performances de plusieurs systèmes de classification on peut par exemple comparer les temps d'exécution, les taux de classification correcte ou d'autres mesures qui caractérisent le processus de classification.

Cette approche basée sur des données "benchmark" présente certaines limites puisqu'un système de classification est en général conçu pour une application particulière et qu'il n'est donc pas très réaliste d'imaginer que des données "benchmark" en adéquation parfaite avec l'application traitée puissent toujours être trouvées. De plus, les critères de performance à prendre en compte diffèrent fortement d'une application à l'autre et chercher un critère universel est sans doute voué à l'échec puisque les comparaisons de performances ne peuvent être gérées indépendamment du contexte de travail. On ne peut donc pas arriver à un degré de généralisation très important, surtout que l'application d'un algorithme à des domaines complètement différents de celui pour lequel il a été conçu risque de conduire à des performances très faibles sur certains critères.

Pour une application donnée, il est cependant nécessaire de mettre en place une méthodologie permettant de garantir une certaine capacité de généralisation du classifieur face à de nouvelles données provenant du système étudié. Deux approches peuvent être évoquées dans ce contexte. La première consiste à réaliser l'apprentissage du système et son évaluation sur un même ensemble de données. Dans cette situation un partage répétitif de l'ensemble de données en deux sous–ensembles (d'apprentissage et d'évaluation) est nécessaire. Différentes méthodologies ont été développées et elles seront décrites dans le paragraphe 1.4.3. La deuxième approche est typique des systèmes pour lesquels l'obtention d'un grand nombre de données étiquetées est possible : un ensemble de données dédié à l'évaluation, indépendant de l'ensemble d'apprentissage, peut alors être fourni et les mesures de performance décrites dans la section suivante peuvent être calculées directement à partir de cet ensemble.

1.4.2 Critères d'évaluation

La mesure de performance la plus simple et qui est à la base de tout autre critère est le taux de bonne classification [5]. Cette mesure est le pourcentage de points de test qui ont été bien classifiés par le système. Elle est donnée par l'équation (1.4), où N_c est le nombre de points de test correctement classifiés et N_t est le nombre de points de l'ensemble de test.

$$T_c = \frac{N_c}{N_t} \times 100\% \tag{1.4}$$

Une manière plus complète et très utilisée pour représenter les performances d'un système de classification est de construire une "matrice de confusion" [99]. Les intitulés de colonne de cette matrice correspondent aux classes d'appartenance déterminées par le système, tandis que ceux des lignes représentent la vraie classe d'appartenance. Ainsi, la valeur N_{ij} trouvée sur la ligne i et colonne j est le nombre de points appartenant à la classe i qui ont été affectés à la classe j, comme illustré

dans le tableau 1.1.

Classe réelle \ Classe prédite	C_0	C_1	$\cdots$	$C_{\|C\|}$
C_0	N_{00}	N_{01}	$\cdots$	$N_{0\|C\|}$
C_1	N_{10}	N_{11}	$\cdots$	$N_{1\|C\|}$
$\vdots$	$\ddots$	$\ddots$	$\ddots$	$\ddots$
$C_{\|C\|}$	$N_{\|C\|0}$	$N_{\|C\|1}$	$\cdots$	$N_{\|C\|\|C\|}$

TABLE 1.1 – Matrice de confusion

Usuellement la matrice de confusion est présentée sous une forme normalisée. Les valeurs de la matrice sont obtenues par la division des valeurs N_{ij} de la matrice du tableau 1.1 par le nombre de points de la ligne i dans l'ensemble de test. Evidemment, pour ce type de matrice de confusion l'évaluation d'un classifieur parfait sera une matrice de confusion identité, comme représenté dans le tableau 1.2.

Classe réelle \ Classe prédite	C_0	C_1	$\cdots$	$C_{\|C\|}$
C_0	1	0	$\cdots$	0
C_1	0	1	$\cdots$	0
$\vdots$	$\ddots$	$\ddots$	$\ddots$	$\ddots$
$C_{\|C\|}$	0	0	$\cdots$	1

TABLE 1.2 – Matrice de confusion normalisée pour un classifieur parfait

Dans le cas de la classification binaire, la matrice a une représentation particulière qui utilise les notations tp, tn, fp et fn : tp ("true positive") est le nombre (ou le pourcentage, selon la représentation) de points correctement classifiés dans la classe recherchée (classe positive), tn ("true negative") est le nombre/pourcentage de points correctement rejetés comme ne faisant pas partie de la classe recherchée, fp ("false positive") est le nombre/pourcentage de points incorrectement placés dans la classe et fn ("false negative") le nombre/pourcentage de points incorrectement rejetés. Cette représentation offre une lisibilité accrue de la qualité du classifieur (cf. tableau 1.3).

C. réelle \ C. prédite	p	n
p	tp	fn
n	fp	tn

TABLE 1.3 – Matrice de confusion pour un classifieur binaire

Pourtant, dans la plupart des applications des mesures plus détaillées sont nécessaires. De telles mesures sont surtout définies dans le cas de la classification binaire, où on doit simplement établir l'appartenance ou la non–appartenance d'une entité à une classe. Quelques notions sont couramment utilisées dans ce contexte. Dans des articles concernant cette problèmatique [143, 120], on trouve la définition des mesures suivantes : l'exactitude, la sensitivité, la spécificité, la précision et la F–mesure. Leurs équations pour le cas binaire sont données par (1.5).

$$\begin{aligned} \text{exactitude} &= \frac{tp+tn}{tp+fn+fp+tn} \\ \text{sensitivité} &= \frac{tp}{tp+fn} \\ \text{spécificité} &= \frac{tn}{fp+tn} \\ \text{précision} &= \frac{tp}{fp+tp} \\ \text{F–mesure} &= \frac{(\beta^2+1)\cdot\text{sensitivité}\cdot\text{précision}}{(\beta^2\cdot\text{précision})+\text{sensitivité}} \end{aligned} \tag{1.5}$$

On remarque que l'exactitude est en fait le taux de classification correcte. La sensitivité est généralement utilisée dans les applications où l'identification des cas positifs est très importante (surtout les applications médicales), la spécificité est plus importante pour les applications où le coût d'une décision "false positive" est très élevé (surtout les applications industrielles en contrôle de qualité). La précision donne une mesure du pouvoir discriminant du classifieur par rapport à la classe de l'application.

Dans la définition de la F–mesure, la valeur β représente l'importance associée à la précision par rapport à l'importance associée à la sensitivité (appelée aussi "rappel"). Une valeur supra–unitaire correspond à une importance supérieure accordée à la précision, alors qu'une valeur sous–unitaire correspond à une importance supérieure associée au rappel. Dans le cas de problèmes où les décisions erronées n'engendrent pas des coûts différents pour les deux classes, la valeur de β est habituellement établie à 1.

Des généralisations de ces mesures sont aussi disponibles pour les problèmes multi–classes [5]. Généralement, la généralisation se fait par une analyse indépendante de chaque classe recherchée, en considérant les autres classes comme une classe générique de rejet. Pour chacune des classes une matrice de confusion binaire peut donc être construite suivant les règles :

- la valeur tp est donnée par le nombre de points appartenant à la classe analysée qui ont été correctement classifiés
- la valeur tn est donnée par le nombre de points appartenant à une des autres classes qui n'ont pas été classifiés comme appartenant à la classe analysée
- la valeur fp est donnée par le nombre de points appartenant à une des autres classes et qui ont été classifiés comme appartenant à la classe analysée
- la valuer fn est donnée par le nombre de points appartenant à la classe analysée et qui ont été classifiés comme appartenant à une des autres classes

Une représentation un peu plus spécialisée est constituée par les courbes ROC (Receiver Operating Characteristic) [114]. Cette représentation est applicable aux problèmes de classification binaire et chaque point de la courbe est en fait une interprétation graphique de la matrice de confusion obtenue pour le même système de classification paramètré différemment. L'avancement sur la courbe se fait par exemple par l'augmentation du nombre de points d'apprentissage utilisés. Les courbes ROC ont surtout été utilisées dans les applications médicales, où il faut décider entre la présence et l'absence d'une certaine maladie. Une courbe ROC représente la relation entre la proportion de "vrai positifs" et la proportion de "faux positifs". Comme les deux notions sont liées aux notions de sensitivité et de spécificité, les courbes ROC sont aussi connues sous le nom de courbes sensitivité vs (1 - spécificité). La courbe obtenue a théoriquement la forme représentée dans la figure 1.8 [62].

La courbe indique le compromis entre la valeur de tp et celle de fp. Elle montre que (pour un classifieur normal) on peut augmenter la valeur de tp en modifiant les paramètres du système vers la valeur maximale (1 pour les représentations normalisées), mais non sans conséquence : l'augmentation de la valeur de fp. La courbe est utilisée afin de choisir les paramètres du système qui offre le meilleur

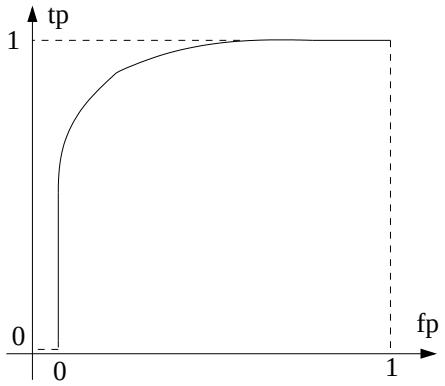


FIGURE 1.8 – Courbe ROC

compromis entre la bonne détection des cas positifs et la mauvaise détection des cas négatifs.

Vapnik & al [151] proposent un autre critère d'évaluation d'un classifieur, basé sur sa "capacité". Les systèmes d'apprentissage de petite capacité sont ceux qui arrivent à avoir un bon taux de classification correcte sur les ensembles de test même à partir d'un ensemble d'apprentissage de faible dimension. Les systèmes de grande capacité sont des systèmes qui offrent un meilleur taux de classification pour les ensembles d'apprentissage de très grande dimension, mais qui ont besoin de beaucoup de points d'apprentissage pour arriver à des valeurs correctes de l'erreur obtenue sur la classification de l'ensemble d'apprentissage lui–même. La dimension VC(Vapnik–Chervonenkis) est justement une manière de mesurer cette capacité.

1.4.3 Méthodes d'évaluation

Ces mesures peuvent être directement calculées si on dispose d'un ensemble de données de test. Malheureusement, dans beaucoup d'applications il est difficile d'obtenir de tels ensembles. C'est pourquoi d'autres méthodes d'évaluation ont été développées. Ces méthodes ont comme principe général le partage (successif) d'un seul ensemble de données disponible en deux sous–ensembles : d'apprentissage et de test. Plusieurs approches sont proposées dans la littérature dont les plus "classiques" sont celles de *cross–validation* et de *bootstrap*.

Le principe de la cross–validation [155] a été mis en place dès les années 60, mais sa mise en œuvre a commencé à être plus populaire avec l'augmentation du pouvoir de calcul des ordinateurs. La méthode est basée sur le ré–échantillonnage de l'ensemble de données sur lequel le système est appris puis testé. L'ensemble de données est séparé en Q sous–ensembles de taille égale. Parmi eux, le système en choisit $Q-1$. Les points contenus dans ces $Q-1$ sous–ensembles sont utilisés pour faire l'apprentissage du système. Ensuite, en utilisant le système appris (règles, équations, réseau neuronal, etc), les points du dernier sous–ensemble (celui qui n'a pas servi à l'apprentissage) sont classifiés. La procédure est répétée pour toutes les permutations possibles, c'est–à–dire Q fois (les Q sous–ensembles ont servi comme ensemble de test une fois). Le taux de classification correcte du système est alors calculé comme la moyenne des taux de classification correcte obtenus sur chaque sous-ensemble de test. Le système cherche à approximer la vraie erreur d'un classifieur, c'est–à–dire l'erreur de classification si le classifieur disposait d'un ensemble d'apprentissage exhaustif. Cette valeur est evaluée avec la méthode donnée en apprenant le système avec plusieurs ensembles d'apprentissage différents. Cela amène à obtenir une représentation de l'erreur estimée comme une fonction caractérisée par une variance et un biais par rapport à la valeur réelle recherchée, en fonction des caractéristiques de l'ensemble d'apprentissage utilisé.

Leave–one–out est un cas particulier de la famille des méthodes de cross–validation. Il consiste à éliminer successivement les exemples disponibles de l'ensemble d'apprentissage. Si l'on considère le cas d'un ensemble d'apprentissage composé de N points, l'apprentissage se fait N fois sur $N-1$ points ($Q = N$: le nombre de sous–ensembles d'apprentissage est égal au nombre de points de l'ensemble de données original). Ensuite le système obtenu est appliqué sur le point qui n'a pas été pris en considération pour l'apprentissage. Le taux de classification correcte est donné dans ce cas particulier de cross–validation par le pourcentage de points qui ont été classifiés correctement. En général la méthode *leave–one–out* est considérée comme n'ayant pas de biais, mais elle a une variance assez importante [19].

Une autre grande famille de méthodes d'évaluation est le *bootstrap*. Elle est recommandée surtout sur les ensembles de données composés de peu de points [155] (par peu de points on entend moins de 30). Dans ces cas–là, la variance a tendance à devenir assez importante, donc une méthode d'évaluation avec une grande variance (comme le leave–one–out) est moins fiable. Le principe, comme pour la cross–validation, consiste à choisir aléatoirement un pourcentage donné des points comme points d'apprentissage et utiliser le reste comme points de test. La différence par rapport à la méthode de cross–validation est le fait que les points qui sont choisis pour l'apprentissage ne sont pas éliminés de l'ensemble original, c'est–à–dire qu'ils peuvent être séléctionés plusieurs fois. Les points qui n'ont pas été choisis comme points d'apprentissage constituent l'ensemble de test. L'opération est répétée plusieurs fois (il est généralement considéré qu'une valeur de 200 itérations est nécessaire afin d'obtenir de bons résultats [155]) et, comme dans le cas de la cross–validation, le taux de classification correcte est la moyenne des taux de classification correcte obtenus sur chaque sous-ensemble de test. Le cas particulier le plus connu de cette approche est la méthode ".632 bootstrap" [47]. Plusieurs comparaisons entre les deux méthodes sont disponibles dans la littérature [4, 91].

1.5 Discussion et conclusion

Dans le contexte décrit dans ce chapitre, quelques problèmes typiques peuvent être identifiés. Ils sont détaillés et analysés dans cette section.

1.5.1 Acquisition des données

La qualité et la pertinence des données d'entrée est un paramètre important qui peut beaucoup influencer les performances de la classification. On doit prendre en considération le fait que les mêmes données acquises à des moments temporels différents peuvent apporter un degré différent d'informations utiles. De plus, même si l'information contenue dans les données est utile, le degré de superposition avec le bruit peut aussi varier avec l'environnement et le vieillissement du dispositif d'acquisition. Il est donc parfois utile pour les systèmes de classification d'avoir un certain degré d'adaptabilité pour permettre de légères variantes dans la manière d'acquérir et de représenter des données.

L'apprentissage actif est un problème lié à l'acquisition de données, lorsqu'il est possible d'influencer sur les données qui seront effectivement collectées. C'est par exemple le cas d'applications d'apprentissage en robotique, où l'équipement hardware peut être lent et plus ou moins imprécis, ce qui peut avoir un effet négatif sur le processus de classification. Il faut aussi prendre en considération les changements qui peuvent intervenir dans le contexte entre le moment d'acquisition des données d'apprentissage et le moment de l'utilisation du système obtenu. Souvent les données d'apprentissage doivent être actualisées périodiquement. Un cas encore plus difficile à gérer est celui où l'équipement

d'exploration peut influencer sur l'ensemble de données collectées. Pour résoudre le problème, il est nécessaire de bien connaître le contexte de l'application et de bien définir les conditions dans lesquelles l'acquisition de données doit être réalisée. On doit aussi choisir les informations qui sont significatives pour l'application et ignorer l'information superflue ou non pertinente. L'apprentissage actif est un problème spécifique aux domaines non encore standardisés, où l'on ne dispose généralement pas d'un nombre suffisant de données de test.

L'apprentissage cumulatif est également un problème lié à l'acquisition de données. Dans de nombreuses applications de classification, une méthode déjà implémentée doit être appliquée sur des données qui n'étaient pas disponibles au moment de l'apprentissage. C'est par exemple souvent le cas des applications médicales, domaine où les bases de données s'enrichissent au cours du temps, induisant une augmentation de la qualité et de la pertinence de l'information contenue. La méthode de classification doit être capable de gérer des données qui ne sont pas obtenues de la même manière que les données sur lesquelles elle a été testée. Le problème consiste donc à rendre le système adaptable aux changements du processus de collecte d'informations. Par exemple, on peut avoir des radiographies de différentes qualités, obtenues par différents types de machines, pré–traitées de différentes manières. Il faut intégrer des données appartenant à un même domaine, mais présentées sous une forme qui diffère.

1.5.2 Supervision et/ou intégration de connaissances

Le principe de base d'un processus de classification donné peut choisir de ne pas tenir compte des informations auxiliaires disponibles sur l'ensemble d'apprentissage. Notamment, pour ce que l'on appelle les systèmes de classification non–supervisés, la classe d'appartenance de chaque exemple d'apprentissage peut être ignorée même si elle est connue. Dans ce cas–là et dans le cas où ces informations ne sont pas disponibles, le classifieur doit réaliser une division naturelle de l'ensemble d'apprentissage dont il dispose. Pour obtenir de bons résultats, on a normalement besoin que les classes à individualiser soient bien séparables. Cette approche est très différente de l'apprentissage supervisé. Une de ses difficultés est la nécessité de trouver de manière automatique les conditions d'arrêt du processus. De plus, si on se situe dans le cas où l'information a priori sur la classe d'appartenance n'existe effectivement pas, il est plus difficile de définir un critère selon lequel on peut évaluer les performances du système d'apprentissage. Dans les systèmes basés sur cette approche, on cite les réseaux neuronaux, la méthode $C - means$ et sa variante floue, etc. En fonction du choix d'utiliser ou non de l'information complémentaire à l'ensemble d'apprentissage, on distingue plusieurs types d'approches pour réaliser l'apprentissage :

- **L'apprentissage utilisant des données étiquetées et non étiquetées**. Il est parfois difficile de disposer de données déjà étiquetées. L'étiquetage est très important pour l'interprétation des sorties de tests pour la méthode choisie par exemple. Le problème qui se pose est donc de savoir si on peut concevoir des algorithmes applicables sur des données non étiquetées pour l'apprentissage de nouveaux concepts.
- **L'apprentissage utilisant des connaissances a priori**. La classification peut être aidée par des informations qui ne sont pas nécessairement contenues dans les données d'apprentissage. Le problème est donc de trouver des arrangements flexibles pour pouvoir insérer ces connaissances a priori, même si elles sont incertaines, abstraites ou symboliques (non numériques).

1.5.3 Taille de l'ensemble de données

Normalement l'étape d'apprentissage consiste en l'analyse et le traitement d'un ensemble d'apprentissage de taille moyenne. Pour chaque type de classifieur la taille idéale de l'ensemble d'apprentissage est individualisée. On peut préciser qu'un ensemble d'apprentissage normal peut avoir entre quelques dizaines et quelques centaines d'exemples d'apprentissage pour chaque classe. En dehors de cet ordre de grandeur, on doit trouver des méthodes adaptées au nombre de données d'apprentissage. Dans le cas des ensembles de données de très grande dimension on peut envisager de faire une sélection des exemples qui peuvent être éliminés sans trop perdre de l'information significative. Cette problématique est brièvement discutée ci–dessous :

- **L'apprentissage en utilisant une base de données très grande :** les bases de données qui sont très grandes et très dynamiques ne peuvent être lues, au mieux, que quelques fois par un ordinateur avec des performances moyennes. Deux exemples typiques sont les bases de données d'Internet ("Web bases") ou les bases des données d'un supermarché. Les méthodes qui supposent parcourir l'ensemble de données plusieurs fois (par exemple les méthodes basées sur la méthode $C - means$) ne peuvent pas être appliquées dans ces situations. Le problème consiste donc à trouver des méthodes qui peuvent aboutir à des résultats pertinents en un seul balayage de l'ensemble de données. Une solution à ce type de problème est le développement d'algorithmes de type data mining [46].
- **L'apprentissage en utilisant une base de données très petite :** les problèmes qui se situent dans ce contexte sont des problèmes spécifiques. Par exemple, on peut avoir un problème de reconnaissance de visages où l'on dispose d'une seule image par individu. Le même problème se retrouve dans l'apprentissage dans le domaine robotique, où le nombre d'exemples pour l'apprentissage est habituellement extrêmement limité. Dans ces cas, si des informations additionnelles ne sont pas disponibles, il faut trouver des méthodes d'apprentissage qui peuvent donner de bons résultats sur un très petit ensemble d'apprentissage.

1.5.4 Relations entre les attributs

Généralement chaque exemple de l'ensemble d'apprentissage est caractérisé par plusieurs attributs qui peuvent être corrélés. Si la corrélation existe, il est fortement probable que l'information offerte par les attributs soit redondante. Il est alors utile de faire une sélection ou une agrégation des attributs. On doit aussi prendre en compte la relation logique entre les attributs, lorsqu'il est possible que la quantité d'informations offerte indépendamment par certains attributs soit plus petite que l'information qui serait offerte si l'on considère l'ensemble de ces attributs. Il faut donc déterminer la nature de la corrélation entre les différents attributs : la relation peut être soit de type cause–effet, soit de type cause commune, soit elle peut indiquer la présence d'un autre phénomène inconnu, à identifier et à classifier. Différents types d'apprentissage prennent en compte ces relations et les traitent afin d'améliorer le résultat final :

- **L'apprentissage avec relations :** pour prendre une décision dans des domaines où la distinction entre les informations pertinentes et celles superflues ou sans importance ne peut pas se faire, il faut rechercher des associations entre les différents attributs disponibles. Par exemple, pour identifier les organisations qui s'occupent de blanchir de l'argent on doit tenir compte de plusieurs informations qui mettent en évidence les liens entre les gens, les organismes, les compagnies et les pays. Le problème est donc de trouver un algorithme qui tient compte des rapports entre plusieurs entités informationnelles pour pouvoir prendre une décision pertinente.
- **L'apprentissage des rapports causaux :** les systèmes d'apprentissage peuvent identifier les

corrélations entre certains événements, mais il est plus difficile d'établir si la corrélation est causée par une liaison de type cause/effet ou si la corrélation est la suite d'une cause commune. Par exemple, il y a un grand degré de corrélation entre le cancer des poumons et les doigts jaunes d'une personne, mais la corrélation est donnée par la cause commune (la personne fume), et il serait inutile d'annuler l'effet des doigts jaunes en ce qui concerne le cancer de poumon, comme montré dans la figure 1.9.

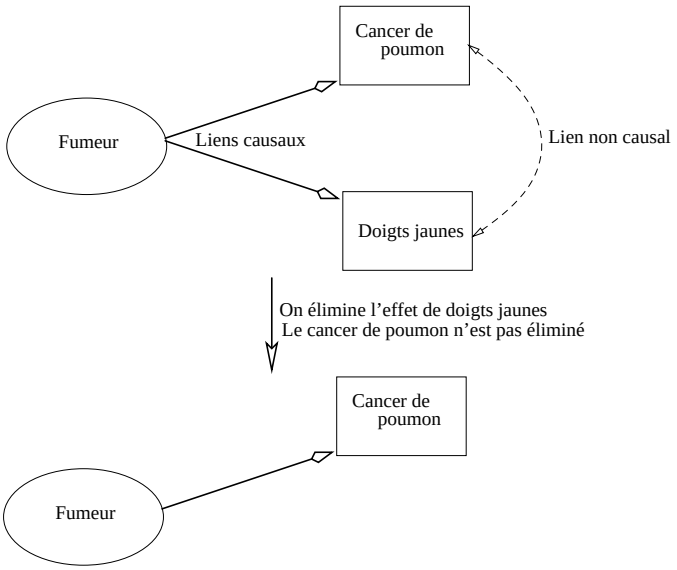


FIGURE 1.9 – Exemple de lien non causal ; en supprimant un effet on ne supprime pas l'autre

1.5.5 Représentation de l'information

Face à la quantité d'informations qui peut caractériser un exemple dans un ensemble de données et à une représentation éventuellement différente des divers attributs, une étape de traitement antérieure et/ou ultérieure à la classification est parfois nécessaire. L'apprentissage multitâche et le traitement des informations représentées sous différentes formes en sont des exemples typiques :

– **L'apprentissage multitâche (multitasking) :** c'est fréquemment le cas dans les applications médicales, où plusieurs symptômes peuvent caractériser le même individu, et dont diverses combinaisons peuvent indiquer différentes maladies. Mais on ne peut pas aborder la classification des données médicales en cherchant directement plusieurs maladies : la séparation entre de très nombreuses classes (maladies) est pratiquement impossible dans la mesure où il est illusoire d'intégrer, dans un unique système, le nombre gigantesque de conditions et de règles qui seraient nécessaires à la description d'un domaine si vaste. Une autre difficulté du domaine médical est le manque de formalisation : l'information disponible n'est pas encore standardisée et structurée de manière consistente. Par conséquent, la classification est habituellement faite pour chaque maladie étudiée. On se situe donc dans le cadre d'une séparation entre deux classes : une maladie particulière existe ou non. Il est donc important d'identifier les attributs qui ne sont utilisables que pour identifier d'autres maladies que celle recherchée afin de les éliminer de l'ensemble d'apprentissage. Différents systèmes de classification peuvent être utilisés afin d'identifier la présence ou l'absence d'une maladie donnée et ensuite un système qui réalise l'intégration de ces classifieurs peut être mis en œuvre. On parle alors de fusion de classifieurs [94]. Le problème consiste à faire le transfert de connaissance entre les systèmes

d'apprentissage. Une illustration d'un tel système est présentée dans la figure 1.10.

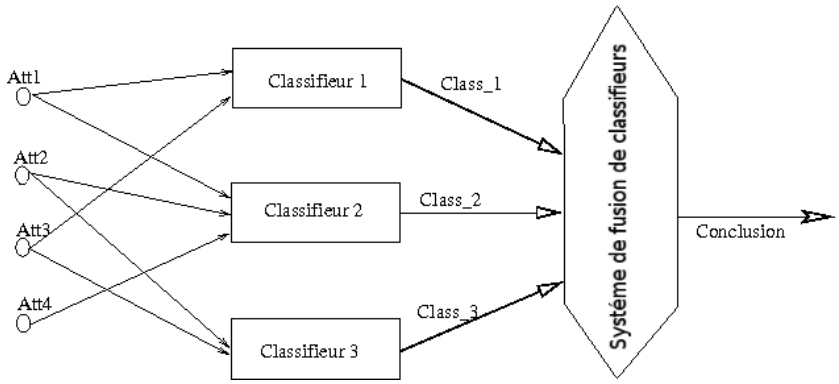


FIGURE 1.10 – Exemple d'architecture multitasking à base de classifieurs binaires

– **L'intégration des types mélangés de données :** l'ensemble de données à utiliser est souvent composé d'attributs de différentes catégories. Par exemple, dans le domaine médical, on peut avoir différents types d'informations : informations numériques (résultats de différents tests, analyses de sang, etc), images (radiographies), données nominales (l'individu est fumeur ou non), etc. Le problème est de savoir s'il est préférable de séparer les algorithmes de classification pour chaque type d'attribut et ensuite de combiner leur résultats grâce à un système de fusion ou d'essayer d'intégrer et de fusionner les différents types d'informations dans une étape d'extraction de caractéristiques.

Un problème particulier dans ce contexte est la visualisation des données multidimensionnelles. La classification peut être souvent aidée par l'utilisateur–expert, mais il est nécessaire de trouver une manière de représenter les données multidimensionnelles afin qu'il puisse les comprendre et les analyser naturellement.

1.5.6 Conclusion

En conclusion, on peut retenir que les systèmes de classification traitent un problème très complexe. Afin de développer un système cohérent, pertinent et qui offre de bons résultats, il faut prendre en compte une multitude d'aspects. Les classifieurs sont généralement dédiés à une application donnée, mais la plupart peuvent être utilisés dans d'autres contextes en respectant les conditions imposées sur le format des données d'entrée et en réglant les paramètres spécifiques.

Avant de présenter le système de classification que cette thèse propose, un deuxième chapitre énumère les principales méthodes de classification présentes dans la littérature ainsi que leur caractéristiques principales, en essayant de positionner la méthode proposée dans le contexte général de la classification de données.

Chapitre 2

Méthodes de classification

Si on revient à la figure 1.3 (page 20), on peut ramener la mise en place d'un système de classification à trois questions (ou problématiques) :

1. Quelles sont les données d'apprentissage ? Généralement c'est l'application en elle–même qui va imposer la réponse en fournissant les données.
2. Quel modèle de représentation choisir ? Il n'y a pas de réponse idéale et c'est au concepteur de déterminer quel pourrait être le modèle le mieux adapté en fonction des caractéristiques de l'application. De plus en plus de travaux cherchent à montrer les équivalences existantes entre les différents modèles de représentation.
3. Quelle méthode d'apprentissage choisir ? La méthode d'apprentissage la plus appropriée reste à définir et c'est certainement sur ce point que le degré de liberté est le plus important.

Afin de présenter les méthodes d'apprentissage sélectionnées, celles–ci sont regroupées en fonction du modèle de représentation qui leur est généralement associé. Ainsi, on distingue trois grandes catégories d'approches : les approches statistiques, les réseaux neuronaux et les systèmes basés sur des règles :

- Les méthodes de classification basées sur l'approche statistique ont une base théorique et pratique très forte.
- Les réseaux de neurones sont une représentation simplifiée du système nerveux biologique. De part leur construction, ils sont très rapides d'un point de vue computationel et possèdent un degré important de parallélisme. Les relations qui se développent au cours de l'apprentissage sont souvent difficiles à comprendre et à suivre. De ce fait, ils sont souvent assimilés à des "boîtes noires".
- Les sytèmes de classification basés sur des règles sont des systèmes qui "apprennent" de manière plus transparente. A partir d'un ensemble de données, ces systèmes construisent des règles de décision qui peuvent ensuite être appliquées sur des exemples inconnus afin d'établir leur classe. Le principal avantage de ce type de classifieur est que l'outil de classification, qui est la règle, peut être exprimé dans un langage naturel et peut donc être compris et interprété par l'utilisateur humain. Ainsi, l'utilisateur du système de classification ne se trouve plus dans la situation de "croire" à un résultat qu'il ne peut pas expliquer, mais au contraire, il peut analyser d'une manière intuitive la démarche de classification et son résultat, sans pour autant avoir de connaissances approfondies dans le domaine de la classification.

D'un point de vue historique, les méthodes présentées dans chacune des trois catégories sont les plus anciennes, mais la plupart des méthodes plus récentes les utilisent comme point de départ.

Actuellement, les principales approches couramment utilisées en pratique sont les méthodes neuronales [66], les méthodes floues [10], différentes combinaisons de ces deux dernières [16] (soit une étape préalable d'implémentation floue pour le calcul des paramètres des réseaux neuronaux, soit l'implémentation d'une classification floue avec une étape initiale de préparation de l'ensemble des données avec des réseaux neuronaux) et des méthodes plus spécifiques, comme le SVM (Support Vector Machine) [28]. Des approches plus récentes, comme les différentes approches de "data mining" [61] sont également de plus en plus fréquemment utilisées, plus particulièrement pour traiter des ensembles de données de très grande taille.

Différentes méthodes de classification peuvent être utilisées soit pour réaliser l'intégralité de la classification, soit pour obtenir des sorties partielles qui servent comme entrées pour un système de fusion d'informations en charge de la classification finale. Dans un premier temps on peut considérer les méthodes indépendamment les unes des autres, mais la tendance générale est de combiner les différentes approches afin d'obtenir de meilleurs résultats. On peut par exemple implémenter un classifieur flou dans le cadre d'une architecture neuronale, ou utiliser la méthode "support vector machine" dans un cadre statistique, etc. Néanmoins, dans les sections suivantes, uniquement les approches "classiques" seront présentées, sachant que toutes les autres approches sont basées sur ces méthodes "de base".

De manière générale, les méthodes présentées dans ce chapitre s'inscrivent dans le domaine de l'apprentissage automatique dont l'objectif est de concevoir des dispositifs qui représentent au mieux le processus qui a généré (ou pourrait avoir généré) les données disponibles. Les méthodes sélectionnées sont toutes relatives à des problèmes d'apprentissage supervisé. Elles ne sont que brièvement décrites, mais plus de détails peuvent être trouvés dans les références bibliographiques indiquées.

2.1 Les approches statistiques

L'apprentissage automatique peut être perçu comme un sous–domaine de la théorie de l'apprentissage statistique qui explore la manière d'estimer des dépendances fonctionelles à partir d'échantillons de données. Dans ce contexte, les principales méthodes (mais aussi les plus anciennes) consistent à estimer les paramètres de modèles particuliers à partir des données disponibles, d'où le nom d'"*inférence paramètrique*" attribué à la discipline dans le cadre des statistiques inférentielles.

2.1.1 Les systèmes de classification bayesiens

De manière générale, l'application de l'analyse statistique de Bayes [7] aux problèmes de classification est basée sur une modélisation probabiliste des classes [21]. En phase d'exploitation, les modèles de classes, étant supposés connus, un classifieur de Bayes détermine la classe d'un exemple $X = [x_1, \ldots, x_n]$ selon l'hypothèse du maximum a posteriori (MAP). Dans ce contexte, la classe d'affectation de X, C_{MAP}, est obtenue par maximisation de la probabilité conditionelle de classe, c'est–à–dire

$$C_{MAP} = argmax_{C_i \in C} P(C_i|X), \tag{2.1}$$

ou encore par application de la règle de Bayes :

$$C_{MAP} = argmax_{C_i \in C} \frac{P(X|C_i) \cdot P(C_i)}{P(X)} = argmax_{C_i \in C} P(X|C_i) \cdot P(C_i), \tag{2.2}$$

où $P(C_i)$ est la probabilité a priori de la classe C_i. Lorsque les classes sont équiprobables, l'hypothèse MAP est similaire à l'hypothèse du maximum de vraisemblance.

La détermination de C_{MAP} selon (2.2) garantit une erreur de classification globale minimale. Malheureusement, les systèmes réels qui sont développés à partir de ce principe ne peuvent pas suivre exactement la théorie, car quelques conditions nécessaires en théorie ne peuvent pas être satisfaites en pratique [132] :

– la connaissance parfaite des statistiques de chaque classe
– la connaissance des probabilités a priori des classes

L'objectif de la phase d'apprentissage d'un classifieur de Bayes est alors d'estimer les probabilités nécessaires à la résolution de (2.2), à savoir $P(X|C_i)$ et $P(C_i)$, à partir de l'ensemble d'exemples disponibles.

Le plus grand problème pour cette approche est le nombre généralement faible d'exemples qui peuvent servir à établir une statistique fiable de chaque classe à représenter. Afin de pouvoir appliquer le principe de Bayes sur un problème réel de classification, quelques hypothèses de simplification sont couramment utilisées. Une de ces hypothèses est l'indépendance conditionelle des différents attributs observés. Si l'on prend en considération deux attributs x_1 et x_2 avec les probabilités conditionelles $P(x_1/C)$ et respectivement $P(x_2/C)$, où $P(A/B)$ est la probabilité de l'événement A, sachant que l'événement B s'est produit et que C est une classe, alors x_1 et x_2 sont probabilistiquement indépendants si la probabilité de leur apparition simultanée est égale au produit des deux probabilités conditionelles :

$$P((x_1, x_2)/C) = P(x_1/C) \cdot P(x_2/C) \tag{2.3}$$

Soit $X = [x_1, x_2, \ldots, x_n]$ l'exemple observé, où n est le nombre d'attributs des objets à classifier. Si l'hypothèse d'indépendance conditionelle est satisfaite, la classe C_{MAP} qui minimise l'erreur de classification pour l'exemple X est donnée par l'équation (2.4).

$$C_{MAP} = argmax_{C_i \in C} \prod_{k=1}^{n} P(x_k/C_i) \cdot P(C_i) \tag{2.4}$$

Pour tout nouvel exemple X, le système de classification, dit classifieur naïf de Bayes, doit alors trouver la classe C_{MAP} qui est la solution de l'équation (2.4) [132]. Le problème principal de cette méthode est l'obtention (ou au moins l'estimation) des probabilités utilisées. Si ces probabilités ne sont pas données a priori il faut les estimer à partir des données disponibles. Ainsi, la probabilité a priori d'une classe, $P(C_i), i \in 1, \cdots, |C|$ est la proportion des points d'apprentissage qui appartiennent à la classe C_i. La probabilité conditionelle $P(x_k/C_i)$ est donnée par la proportion de l'événement x_k parmi tous les exemples qui composent la classe C_i.

Cette méthode de classification donne de bons résultats si l'hypothèse d'indépendance conditionelle est raisonnablement correcte. Des variantes moins naïves mais plus complexes prennent en compte des degrés de dépendance plus ou moins élevés entre les attributs et conduisent à l'organisation de l'ensemble des attributs en un réseau bayesien [22].

2.1.2 Les discriminants linéaires – LDA (Linear Discriminant Analysis)

Les systèmes de classification basés sur une analyse linéaire discriminante (LDA) reposent tout comme les classifieurs de Bayes sur une représentation probabiliste de l'information et sur la règle

d'affectation à la classe MAP. Ce sont en fait les hypothèses sous–jacentes à la résolution pratique de l'équation 2.2 qui conduisent à des méthodes d'orientation très différente.

Ainsi, l'indépendance des attributs est l'hypothèse de travail des classifieurs naïfs de Bayes alors que l'approche par analyse linéaire discriminante est basée sur les hypothèses de multinormalité de la vraisemblance des classes et d'homogénéïté de leur matrice de variance–covariance (hypothèse d'homoscédasticité). Selon ces hypothèses, $P(X|C_i)$ suit une loi gaussienne multidimensionnelle $N(\mu_i, \Sigma_i)$ et $\Sigma_i = \Sigma, i = \overline{1, |C|}$. Dans ce cas, le score discriminant de chaque classe s'exprime linéairement par rapport aux attributs et l'exploitation de la règle de Bayes conduit à l'implémentation d'un séparateur linéaire.

Finalement, le principe des systèmes discriminants linéaires est donc assez simple. Il consiste à trouver les équations linéaires qui définissent les fonctions de classement à chacune des classes. Celles–ci sont de la forme (2.5) [139], où les x_i sont les valeurs du vecteur d'observations à classifier et les w_i les poids qui doivent être déterminés à partir des données d'apprentissage.

$$w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_n \cdot x_n + w_0 \tag{2.5}$$

On peut donc considérer un système de classification LDA comme un système paramétrique évaluant pour chaque classe une simple somme pondérée des valeurs du vecteur observé. La classe qui maximise la fonction de classement (2.5) est la classe qui sera sélectionnée.

Si le problème de classification consiste à séparer deux classes et que les objets à classifier sont caractérisés par n attributs, la comparaison des deux fonctions de classement conduit à une inégalité linéaire qui définit un hyper–plan séparateur $(n-1)$–dimensionnel. Pour $n = 3$, la surface de séparation est un plan et pour $n = 2$ c'est une simple droite, comme montré dans la figure 2.1. Dans le cas binaire, où seulement deux classes doivent être séparées, un seul hyper–plan de séparation est nécessaire. S'il faut faire une distinction entre plusieurs classes, une surface linéaire $(n-1)$–dimensionnelle doit être calculée pour chaque classe. Ces surfaces sont ensuite agrégées afin d'obtenir les surfaces séparatrices finales.

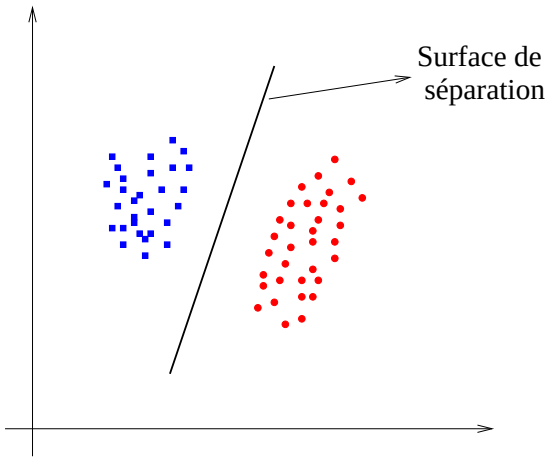


FIGURE 2.1 – Exemple de surface de séparation linéaire

Par contre, dans la plupart des cas réels de classification, les classes se superposent, ce qui rend impossible leur séparation par une simple surface linéaire. Afin de résoudre cette problématique, il peut être utile de construire des surfaces plus complexes, par exemple par analyse discriminante quadratique (hypothèse d'homoscédaticité levée). A noter que comme toute surface courbe peut être

approximée par une succession de surfaces linéaires, il est possible de réduire ce type de problèmes à des problèmes linéaires par morceaux.

2.1.3 La méthode des plus proches voisins

La méthode des k plus proches voisins, "kppv" ou encore "kNN" (k Nearest Neighbours), est l'une des premières méthodes non–paramétriques de classification développées [27]. Son principe de base est très intuitif : un exemple à classer sera placé dans la classe où se trouve la majorité des exemples connus dans son voisinage.

Cette méthode, complètement non–paramétrique, n'effectue aucune hypothèse sur la distribution des nuages de points, mais repose sur une estimation locale des probabilités au voisinage du point à classer. La principale difficulté est alors de définir de manière adéquate le voisinage. D'un point de vue géométrique, la surface de séparation entre les classes n'a donc pas de forme prédéfinie. Selon les exemples observés elle peut alors prendre des formes aléatoires très complexes.

A l'arrivée d'un exemple à classer, le système le compare avec les exemples de l'ensemble d'apprentissage et cherche ses k plus proches voisins, c'est–à–dire les points d'apprentissage ayant, dans l'espace des attributs, les k plus faibles distances au point à classer. Ensuite, le système de classification choisit comme classe de sortie la classe majoritaire parmi les k plus proches voisins identifiés.

Le prérequis à toute recherche de voisins nécessite que soit définie une distance entre exemples. Sachant qu'un exemple est en fait un vecteur d'attributs, la similarité de deux vecteurs est mesurable par la distance entre les vecteurs. Plusieurs types de distances vectorielles peuvent être énumérées : la distance absolue, la distance Euclidienne, différentes distances normalisées, etc. Typiquement, les distances normalisées sont utilisées afin de pondérer les différents attributs du vecteur.

Cette méthode donne de bons résultats si les attributs utilisés sont pertinents, mais les performances diminuent fortement si les attributs sont redondants ou non–pertinents. Le caractère non–paramétrique des classifieurs kppv leur confère la particularité d'avoir une phase d'apprentissage pratiquement inexistante, puisque celle–ci se résume au stockage des données d'apprentissage. Dans ce type d'approche souvent qualifiée d'apprentissage paresseux (lazy learning), aucune généralisation des données d'apprentissage n'est réalisée avant l'utilisation du classifieur. Ainsi, c'est en phase d'exploitation du classifieur kppv et à chaque nouvelle demande de classement que la recherche des k plus proches voisins doit être effectuée. L'optimisation de ce processus couteux en temps de calcul, notamment lorsque la taille de l'ensemble d'apprentissage est élevée, est l'objet de nombreux algorithmes exploitant des structures de données appropriées.

2.2 Les approches neuronales

De nombreux réseaux de neurones sont connus et exploités actuellement, mais les principes basiques peuvent être illustrés sur deux concepts "classiques", les perceptrons et les réseaux multi–couches.

2.2.1 Les perceptrons linéaires

Le problème de classification le plus simple consiste à choisir parmi deux classes (C_0 et C_1). Le perceptron est un système qui choisit une de ces deux classes comme classe d'appartenance d'un

objet–entrée. Dans ce cas, le perceptron implémente en fait un discriminant linéaire et construit la ligne de séparation donnée par l'équation (2.5). Le principe de fonctionnement du perceptron linéaire est illustré dans la figure 2.2. Les $x_i, i = \overline{1,n}$ sont les n composantes de l'exemple analysé et les pondérations $w_i, i = \overline{1,n}$ sont les coefficients qui définissent la ligne de séparation et qui doivent être calculés. Le coefficient w_0 s'appelle "seuil" ou "biais" du perceptron [155].

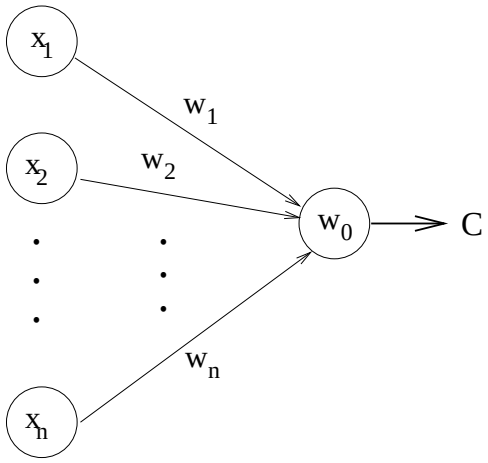


FIGURE 2.2 – Principe du perceptron linéaire

La sortie du perceptron sera alors donnée par l'équation (2.6). Afin d'allouer l'exemple à la classe C_1 il faut que la somme pondérée des attributs soit supérieure à la valeur $-w_0$.

$$C = \begin{cases} C_1 & si \quad \sum_{i=1}^{n} w_i \cdot x_i + w_0 > 0 \\ C_0 & autrement \end{cases} \tag{2.6}$$

De même que pour les discriminants linéaires, la tâche la plus importante et qui définit le perceptron est le calcul des pondérations $w_i, i = \overline{0,n}$. La mise en œuvre du perceptron contient une étape d'apprentissage, lorsque des exemples connus sont utilisés pour déterminer les valeurs des poids w_i. Si le principe de classification est identique à la méthode des discriminants linéaires, le principe de l'apprentissage est très différent : les statistiques de l'ensemble d'apprentissage sont complètement ignorées. Le principe est davantage basé sur un apprentissage séquentiel et répétitif. Les objets d'apprentissage sont présentés au système selon un ordre prédéfini et les pondérations sont modifiées afin de corriger les erreurs de sortie. Si la sortie est correcte, les pondérations ne changent pas. La procédure d'ajustement des pondérations est présentée dans 2.1. Typiquement, le cycle des opérations 1 - 3 s'appelle "epoch" d'apprentissage.

Les pondérations w_i sont initialisées de manière aléatoire (tyiquement avec des valeurs $\in [0,1]$). Si l'on note les pondérations courantes $w_i(t)$, où t est un numéro utilisé afin d'identifier l'itération, le calcul des pondérations actualisées $w_i(t+1)$ se fait conformément à l'équation (2.7). La tâche du perceptron est alors de déterminer pour chaque itération la valeur Δw_i.

$$\begin{aligned} w_0(t+1) &= w_0(t) + \Delta w_0(t) \\ w_i(t+1) &= w_i(t) + \Delta w_i(t), \ \forall i = \overline{1,n} \end{aligned} \tag{2.7}$$

Si l'on considére un problème de classification binaire, une manière assez simple de construire le Δw_i est donnée par l'équation (2.8), où s est l'indice de la classe de sortie du perceptron et v la vraie

Début
1. Faire entrer un nouvel exemple
2. Si la classe de sortie C_s est différente de la classe correcte d'appartenance C_v recalculer les pondérations de chaque composante
3. Répéter 1 et 2 jusqu'à la fin de l'ensemble d'apprentissage
4. Si l'erreur globale obtenue pour l'ensemble des points d'apprentissage est supérieure à un seuil, répéter 1 – 3.

Fin

Procédure 2.1 – Procédure d'apprentissage pour un perceptron linéaire

classe d'appartenance de l'exemple. Par exemple, dans le cas d'une classification "false positive", où le système classifie de manière erronée l'exemple dans la classe C_1 ($s = 1, v = 0$), le biais est diminué d'une unité et les pondérations décrémentées proportionnellement à la valeur de l'attribut concerné. De même dans le cas d'une classification "false negative" ($s = 0, v = 1$), le biais est augmenté d'une unité et les valeurs des pondérations en proportion des valeurs d'attributs.

$$\begin{aligned} \Delta w_0(t) &= v - s \\ \Delta w_i(t) &= (v - s) \cdot x_i, \ \forall i = \overline{1, n} \end{aligned} \tag{2.8}$$

Un problème important pour l'apprentissage d'un perceptron linéaire est la convergence de l'algorithme. La théorie de la convergence du perceptron [8] montre que si les points d'apprentissage définissent des classes complètement séparables à l'aide d'une surface linéaire, le perceptron converge après un nombre suffisamment élevé d'itérations. Pourtant, le temps nécessaire pour obtenir cette convergence peut être très significatif. Afin d'éviter cet inconvénient, quelques observations peuvent être prises en considération :

– la normalisation des données d'entrée : selon la signification des attributs, ils peuvent couvrir des gammes de valeurs très differentes, ce qui peut beaucoup influencer le temps de convergence. Une technique assez répandue est de ramener tous les attributs dans un intervalle fixé (typiquement $[0, 1]$).
– l'introduction d'un paramètre contrôlé dans l'ajustement de $\Delta w_i(t)$: typiquement la valeur de ce paramètre diminue dans le temps, avec l'itération. Ce paramètre augmente la vitesse de convergence et la stabilité de l'étape d'apprentissage.

Pour le cas des classes non–séparables par une surface linéaire, des variantes, comme les systèmes d'apprentissage de type LMS (Least Mean Square) ont été proposés [119]. Ces systèmes reposent sur le fait que la sortie ne sera pas une classe, donc une décision claire, mais des "degrés d'appartenance" de l'exemple à chacune des classes apprises. La structure de base, présentée dans la figure 2.2, est alors légèrement modifiée, comme montré dans la figure 2.3. Chaque sortie du système est un "degré d'appartenance" à une des classes apprises ($m = |C| - 1$). Pour des raisons de simplicité de la notation les biais associés à chaque classe sont notés b_c à la place de w_{0c}. Les algorithmes d'apprentissage et d'utilisation sont simplement une répétition pour chaque classe des algorithmes décrits pour le perceptron linéaire. La sortie de chaque neurone, qui donne un "degré d'appartenance" à la classe correspondante, s'appelle "activation" du neurone. Elle est obtenue en éliminant la phase de seuillage final (2.6), comme montré dans l'équation (2.9).

$$A = \sum_{i=1}^{n} w_i x_i + w_0 \tag{2.9}$$

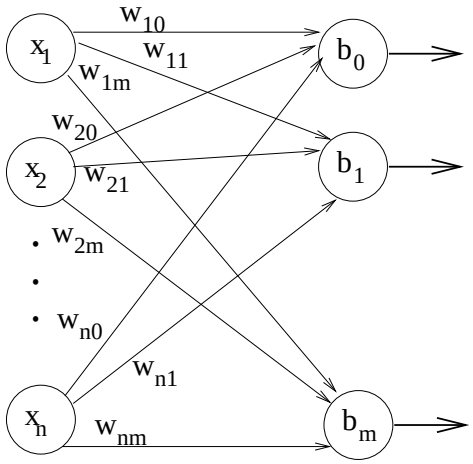


FIGURE 2.3 – Principe du perceptron linéaire LMS

Ce modèle sera utilisé de suite pour expliquer brièvement les réseaux de neurones multi–couches.

2.2.2 Les réseaux multi–couches

Pour les problèmes de classification plus complexes, où la précision de la classification est très importante, les simples perceptrons ne peuvent pas donner des résultats satisfaisants, d'où la nécessité de construire des systèmes plus évolués [119].

Le principe de base des réseaux de neurones multi–couches est de combiner plusieurs perceptrons afin de raffiner la sortie. La combinaison la plus intuitive est de fournir les activations des neurones d'une première couche comme entrées d'une deuxième et ainsi suite. Le principe général est présenté dans la figure 2.4. Si on prend en considération la figure 2.3, une couche intermédiaire de neurones, caractérisés par le biais $s_i, i = \overline{0, p}$, est intercalée entre les entrées et la couche de sortie. Les activations des neurones de la couche intermédiaire (habituellemnt appellée "couche cachée" – hidden layout) forment les entrées de la couche finale. Etant donné que toutes les entrées sont connectées à tous les neurones cachés et que toutes les sorties des neurones cachés sont connectées à tous les neurones de la couche finale, ce type de réseau porte le nom de réseau complètement connecté.

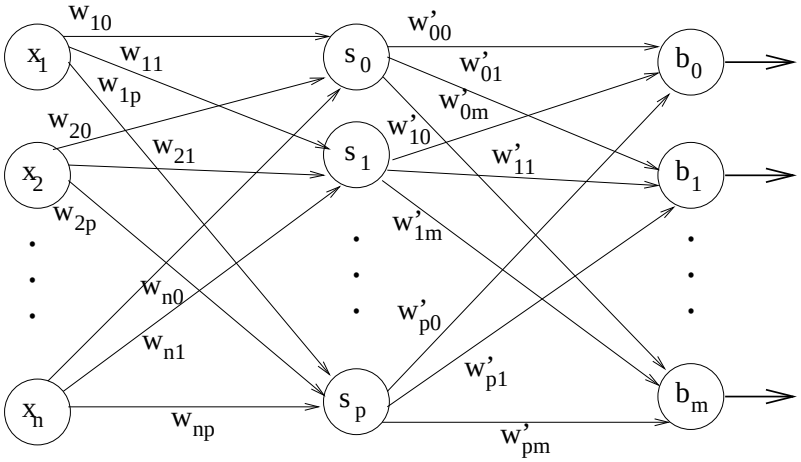


FIGURE 2.4 – Principe des réseaux de neurones multi–couches

Un réseau neuronal comme celui présenté dans la figure 2.4, avec une seule couche cachée, est

généralement suffisamment puissant pour pouvoir décrire n'importe quel type de surface de séparation entre les classes si le nombre de neurones cachés est assez important. Même si l'ajout de couches intermédiaires peut raffiner les résultats obtenus, il est généralement considéré qu'un réseau avec une seule couche intermédiaire a approximativement le même pouvoir discriminant [8].

Pour les réseaux à une seule couche, l'activation d'un neurone était donnée par une simple combinaison linéaire des entrées selon l'équation (2.9). Pour le cas multi–couches, l'activation est calculée d'une manière plus complexe (non linéaire) de façon à augmenter la représentativité du réseau et ne plus être limité à des surfaces de séparation linéaires. Soit le neurone j, situé dans une couche quelconque du réseau. La valeur d'entrée caractéristique à ce neurone, In_j, est donnée par la somme entre le biais qui lui correspond et les sorties pondérées de toutes les unités de la couche antérieure auxquelles il est connecté. Les unités de la couche antérieure peuvent être des neurones cachés ou des attributs de l'exemple à analyser. L'activation du neurone j, A_j, est une fonction qui dépend de cette valeur d'entrée. Généralement la fonction qui donne cette dépendance est une fonction sigmoïdale, comme montré dans l'équation (2.10).

$$A_j = \frac{1}{1 + x^{-In_j}} \tag{2.10}$$

L'apprentissage du réseau multi–couches se fait, comme pour le cas du perceptron linéaire, en plusieurs "epochs". Chaque "epoch" consiste à analyser successivement tous les vecteurs d'entrée qui forment l'ensemble d'apprentissage et à raffiner les poids des différents neurones des différentes couches si l'activation des neurones de la couche finale n'est pas celle attendue. L'ajustement des poids se fait selon un principe similaire à celui présenté pour le perceptron linéaire. Cependant, la non–connaissance des "vraies" sorties pour les couches cachées nécessite d'avoir recours à des techniques algorithmiques plus évoluées comme par exemple de l'algorithme de rétropropagation du gradient couramment utilisé.

Pour l'étape d'apprentissage, un problème important est le choix de la condition d'arrêt. Habituellement deux approches sont utilisées séparément ou de manière combinée :

- nombre d'epochs fixé : après un nombre pré–déterminé d'epochs, le processus d'apprentissage s'arrête et les poids obtenus sont utilisés pour la classification des objets inconnus. Typiquement cette valeur est de l'ordre de quelques milliers d'epochs.
- évolution de l'erreur globale : on choisit un nombre d'epochs N pour évaluer l'erreur globale. Si l'erreur globale pour les N plus récentes epochs n'est pas plus petite que pour les N epochs antérieures, l'apprentissage est arrêté, en considérant que le système est arrivé à saturation. Typiquement on mesure l'erreur globale sur quelques centaines d'epochs.

Un réseau de neurones peut garantir une erreur de classification nulle pour tout ensemble d'apprentissage cohérent si le nombre de neurones dans les couches cachées est assez important. Par ensemble d'apprentissage cohérent, on entend un ensemble qui ne contient pas des objets identiques associés à des classes différentes. Par contre, l'augmentation du nombre des neurones peut produire une sur–spécialisation du système sur les exemples de l'ensemble d'apprentissge, donc peut avoir une influence négative sur sa capacité de généralisation. De plus, l'augmentation du nombre de neurones du réseau mène à une augmentation significative de la durée du processus d'apprentissage. Typiquement, on considère qu'un nombre de 10 neurones dans les couches cachées est suffisant pour obtenir de bons résultats sur l'ensemble d'apprentissage mais aussi sur des objets inconnus.

2.3 Les approches basées sur des règles

Toutes les méthodes présentées dans les sections précédentes ont deux désavantages de base :

- elles travaillent exclusivement avec des attributs numériques ou qui peuvent être facilement implantés dans le monde numérique en gardant toute leur signification,
- elles ne sont pas intuitives et ne peuvent pas être comprises par un utilisateur qui n'est pas familiarisé avec l'environnement mathématique scientifique.

Parmi les méthodes de classification basées sur des règles, on peut en identifier certaines qui permettent de travailler avec des entités linguistiques. De plus, tous les systèmes de cette catégorie produisent des résultats compréhensibles par un utilisateur expert dans le domaine de l'application considérée, mais pas forcément expert dans le domaine de la classification.

Généralement une règle de classification a la forme "**Si** X **et** Y **et** $\cdots$ **alors** classe est C_i" [17]. On remarque qu'une règle est formée de plusieurs composantes :

- la prémisse : c'est la partie de la règle située entre le **Si** et le **Alors**. Elle est composée de plusieurs "propositions" élémentaires (X, Y, etc.) connectées entre elles par des opérateurs logiques (et, ou). Chaque proposition est interprétée comme "vraie" ou "fausse" d'un point de vue logique, l'évaluation globale de la prémisse étant réalisée selon les connecteurs.
- la conclusion : si la prémisse est évaluée comme "vraie", une décision sur la classe d'appartenance peut être prise (avec la règle proposée, la décision est que l'objet appartient à la classe C_i).

La plupart des systèmes de classification basés sur des règles associe une classe avec une règle, c'est–à–dire que la prémisse de chaque règle définit complètement les conditions que doit satisfaire un objet pour appartenir à la classe concernée.

2.3.1 Les arbres de décision

Les arbres de décision sont une manière de répartir des objets dans différentes catégories, selon différents critères. Sans entrer dans la théorie des graphes et des arbres [37], quelques notions vont être introduites lorsqu'elles sont nécessaires pour la compréhension de cette approche.

Un exemple typique d'arbre est l'arborescence d'un dossier informatique "home" stocké sur un disque dur, comme présenté dans la figure 2.5. Un arbre est composé en général de "nœuds", "branches" et "feuilles". Un nœud est une entité qui donne l'accés à une ou plusieurs branches. Une branche contient des nœuds et aboutit à un nœud final, appelé feuille. Dans la figure 2.5, les dossiers "home" et "work" sont des nœuds de l'arbre présenté. Le nœud "home" conduit à la branche qui contient le fichier "Screenshot.jpg" (une feuille) et à la branche qui contient le nœud "work" avec ses trois branches–feuilles "thèse.*". Le nœud "home", qui est "l'origine" de l'arbre, s'appelle la "racine" (root) de l'arbre. Comme règle générale, un arbre ne peut pas contenir de boucles, et une seule branche peut "entrer" dans un nœud.

L'utilisation de la théorie des arbres pour construire un système de classification se fait en respectant quelques règles de base [129, 124, 155] :

- Chaque nœud de l'arbre contient une condition binaire (qui peut être vraie ou fausse) ou une comparaison d'un attribut de l'objet avec des valeurs–clé. Les arbres dont les nœuds contiennent uniquement des conditions binaires s'appellent "arbres binaires".
- Chaque feuille de l'arbre correspond à une décision. Généralement, une feuille est associée à une seule classe, mais une classe peut être associée à plusieurs feuilles.

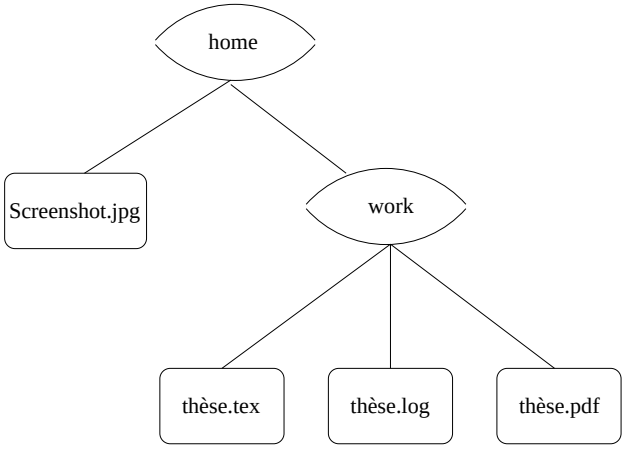
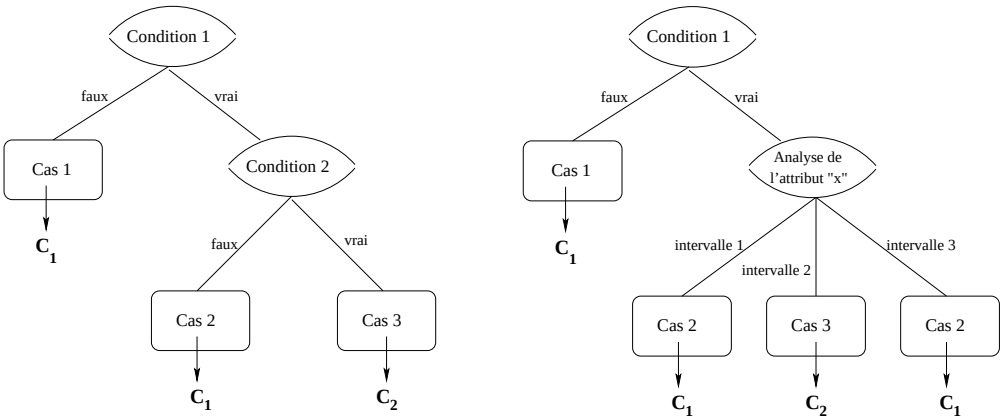


FIGURE 2.5 – Exemple d'arbre

La figure 2.6 présente deux exemples d'arbres de décision. L'arbre de décision dans la figure 2.6(a) est un arbre binaire : chaque nœud a exactement deux branches qui correspondent aux deux valeurs de vérité possibles, "vrai" et "faux", pour les conditions 1 et 2. L'arbre 2.6(b) est un arbre de décision général : ses nœuds contiennent une condition binaire (condition 1), mais aussi une condition générale relative à l'appartenance de la valeur de l'attribut x à l'un des intervalles 1, 2 ou 3. Comme règle générale, les branches des nœuds contenant des conditions non–binaires doivent couvrir tout le domaine de définition (plus exactement toutes les valeurs possibles de l'attribut analysé) sans se superposer. Par exemple, si l'attribut $x \in [0, 3)$, les trois intervalles peuvent être $[0, 1), [1, 2)$ et $[2, 3)$. Une superpostion des intervalles pourrait mener à une ambiguïté dans la décision finale. A l'inverse, si les valeurs possibles n'étaient pas toutes couvertes, le système pourrait échouer dans sa tâche et se retrouver dans une situation d'indécision.



(a) Arbre de décision binaire

(b) Arbre de décision général

FIGURE 2.6 – Exemples d'arbres de décision

Un chemin qui mène de la racine à une feuille correspond à une règle conjonctive (la condition de chacun des nœuds du chemin est évaluée et la suite du parcours est choisie selon le résultat). Si plusieurs feuilles correspondent à la même classe, les règles conjonctives qui correspondent aux chemins qui mènent à ces feuilles sont agrégés de manière disjonctive (il suffit d'avoir un chemin qui

est satisfait pour obtenir la classe).

Par exemple dans le cas de l'arbre binaire (cf. figure 2.6(a)), la classe C_2 est mise en correspondance avec la règle conjonctive : "**Si** Condition 1 **et** Condition 2 **alors** C_2". Par contre, la classe C_1 est mise en correspondance avec une règle qui agrége disjonctivement deux règles conjonctives : "**Si** non(Condition 1) **ou** (Condition 1 **et** non(Condition 2)) **alors** C_1"

Un arbre de décision idéal est le plus petit arbre qui donne une erreur de classification nulle. Même s'il est possible de construire un arbre qui obtient un taux de classification correcte de 100% pour les exemples d'apprentissage, il est impossible de garantir que le même arbre va classifier correctement des objets inconus. En effet, comme pour les méthodes précédentes, une sur–spécialisation du classifieur sur l'ensemble d'apprentissage peut apparaître. Le but de l'apprentissage est donc de trouver l'arbre de la plus petite dimension qui génère de bons résultats sur l'ensemble d'apprentissage et qui a un bon pouvoir de généralisation.

La manière la plus simple d'apprendre un arbre de décision est de choisir un attribut et de construire la condition qui individualise au mieux les classes en ne prenant en compte que cet attribut. Cette condition est placée à la racine de l'arbre. Sur chaque branche (qui correspond à un intervalle pour un attribut numérique, à un ensemble de valeurs possibles pour un attribut symbolique ou encore à la valeur de vérité d'un attribut booléen), on continue par :

- ajouter une feuille qui correspond à une décision si le chemin parcouru est suffisant pour pouvoir prendre une décision pertinente
- ajouter un nœud contenant une condition sur un nouvel attribut si une décision ne peut pas être prise

Ce processus est répété jusqu'à ce que toutes les branches finissent par des feuilles (décisions). A noter que cette situation doit être obtenue avant l'épuisement de l'ensemble des attributs disponibles.

Il est intéressant de remarquer que la dimension de l'arbre, donc la vitesse de classification ainsi que la vitesse d'apprentissage, dépend beaucoup de l'ordre choisi pour analyser les attributs. Plus les attributs pertinents sont analysés tôt, plus la profondeur de l'arbre peut diminuer. L'ordre des attributs (donné par leur pertinence) peut être dicté par des connaissances a priori ou par des analyses spécialisées [124].

2.3.2 Règles en format linguistique

La transposition des règles qui correspondent aux arbres de décision dans un format logique les rend potentiellement plus puissantes. Ce format permet une relaxation de l'exclusivité mutuelle des règles (des cas identiques – valeurs identiques des attributs analysés – peuvent appartenir à des règles différentes). Si l'on prend le cas de l'arbre de décision présenté dans la figure 2.6(a), l'ensemble de règles qui correspond à cet arbre est donné par (2.11).

$$\begin{array}{c} \textbf{Si } \text{non(Condition 1) } \textbf{alors } C_1 \\ \textbf{Si } \text{Condition 1 } \textbf{et } \text{Condition 2 } \textbf{alors } C_2 \\ \textbf{Si } \text{Condition 1 } \textbf{et } \text{non(Condition 2) } \textbf{alors } C_1 \end{array} \tag{2.11}$$

En analysant l'ensemble d'apprentissage, on peut éventuellement se rendre compte que la troisième règle de (2.11) pourrait être remplacée par une règle qui s'appuie sur une condition plus simple, ce qui transforme l'ensemble des règles en (2.12).

$$\begin{array}{c} \textbf{Si } \text{non(Condition 1) } \textbf{alors } C_1 \\ \textbf{Si } \text{Condition 1 } \textbf{et } \text{Condition 2 } \textbf{alors } C_2 \\ \textbf{Si } \text{Condition 3 } \textbf{alors } C_1 \end{array} \tag{2.12}$$

La première et la dernière règle de l'ensemble (2.12) peuvent être satisfaites simultanément, sans que se pose un problème de superposition des classes, car les deux règles correspondent à la même décision. Par contre, la dernière règle peut éventuellement être satisfaite en même temps que la deuxième règle, ce qui peut produire des ambiguïtés dans l'espace de sortie (un même objet peut être alloué à la fois à la classe C_1 et C_2). Même si aucune ambiguïté n'apparaît, dans le cas où la dernière règle n'a aucun attribut commun avec les deux autres, il est impossible de créer un arbre qui implémente exactement l'ensemble de règles (2.12).

Cette manière de construire les règles offre plus de flexibilité, mais le désavantage principal est la possibilité d'apparition d'effets d'interaction inattendus entre les règles, à cause justement du manque d'exclusivité mutuelle. Par conséquent, l'apprentissage de ce type de règles est un point très sensible. Habituellement, l'apprentissage consiste en une analyse complète et complexe de l'ensemble d'apprentissage [155] afin de trouver les règles les plus simples, courtes et efficaces. Le processus d'apprentissage peut donc devenir très long et consommateur de temps. Par contre, si les règles construites sont très synthétiques, le résultat de la classification elle–même peut être obtenu très rapidement.

2.4 Les approches basées sur des règles floues

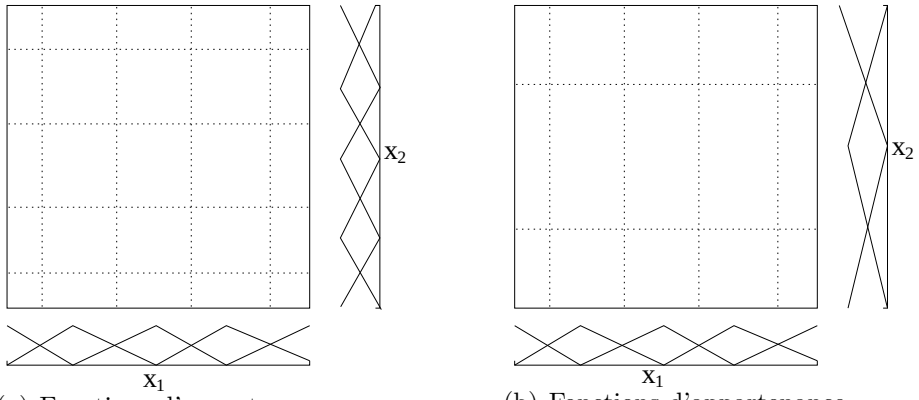
Si les systèmes de classification aujourd'hui utilisés sont fréquemment construits à partir de règles floues, pour la flexibilité qu'elles apportent par rapport aux règles linguistiques conventionelles, il est fait un usage exclusif d'une famille particulière de règles floues, dites règles floues conjonctives. Ces dernières sont présentées dans une première partie de ce paragraphe, qui détaille à la fois leur utilisation et leur apprentissage. La méthode de classification proposée dans cette thèse, qui sera décrite dans les chapitres 3 et 4, est en fait basée sur une autre famille de règles floues, dites règles graduelles, également présentées en fin de ce chapitre.

2.4.1 Règles floues conjonctives dans le domaine de la classification

La notion de "règle floue" a été introduite pour la première fois par Zadeh [159] en 1965. Depuis, les applications sont devenues de plus en plus nombreuses, notamment dans le domaine du contrôle automatique. Le principe de base de ces règles est identique à celui des règles linguistiques présentées dans la section 2.3.2. Elles reposent sur la transposition des connaissances exprimées en langage naturel dans des règles du type "**Si** ... **alors**..." interprétables par les systèmes automatiques et informatiques [98, 127]. Leur spécificité est de permettre une interprétation nuancée de la vérité d'une proposition qui n'est alors plus forcément soit vraie soit fausse, mais est qualifiée par un degré de vérité entre 0 (faux) et 1 (vrai).

Les règles floues ont été d'abord utilisées dans le contrôle automatique et leur utilisation comme règles de classification n'a pas été immédiate [78]. C'est Ishibuchi [76, 77] qui propose une formalisation qui s'oriente dans cette direction. Dans [76], Ishibuchi propose une méthode pour générer des règles floues de classification à partir de données réelles dans un espace 2D en utilisant une grille floue simple, comme montrée dans la figure 2.7(a). Les fonctions d'appartenance définies pour les

deux attributs sont identiques, ce qui conduit à un découpage de l'espace de représentation 2D en carrés et est donc assez limitatif. En utilisant des fonctions d'appartenance indépendantes sur les deux axes, on arrive à représenter des rectangles dans le domaine des attributs, comme montré dans la figure 2.7(b).



(a) Fonctions d'appartenance identiques sur les deux axes

(b) Fonctions d'appartenance indépendantes sur les deux axes

FIGURE 2.7 – Exemples de grilles floues simples

La classification basée sur des règles floues conjonctives se fait en quelques étapes, décrites dans [78] : formuler le problème de classification, réaliser la partition floue, générer les règles floues et enfin classifier des entrées inconnues. Afin de présenter ces étapes, l'argumentation de [78] est suivie.

Dans le but d'illustrer le processus, on se restreint à un problème de classification dans l'espace 2D $[0,1] \times [0,1]$. Afin de réaliser la partition floue de l'espace considéré, on divise chacun des deux axes en I, (respectivement J) sous–ensembles flous $\{A_1^I, \ldots, A_I^I\}$, (respectivement $\{B_1^J, \ldots, B_J^J\}$). Ces sous–ensembles peuvent être caractérisés par n'importe quel type de fonction d'appartenance : triangulaire, trapézoïdale ou bien exponentielle. Si on considère le cas des fonctions d'appartenance triangulaires, on obtient comme fonction d'appartenance a_i^I associée à A_i^I la fonction donnée par l'équation (2.13) et par analogie pour B_i^J l'équation (2.14).

$$a_i^I(x_1) = \max(1 - \frac{|x_1 - \frac{i-1}{I-1}|}{\frac{1}{I-1}}, 0) \tag{2.13}$$

$$b_j^J(x_2) = \max(1 - \frac{|x_2 - \frac{j-1}{J-1}|}{\frac{1}{J-1}}, 0) \tag{2.14}$$

Ainsi, l'espace 2D considéré est partagé en IJ régions rectangulaires, chaque région étant associée à une classe. Le principe peut être facilement généralisé pour un espace N–dimensionnel. Pour un espace $3D$ on "découpe" en parallélépipèdes et pour $N > 3$ on peut imaginer des structures avec les mêmes propriétés.

Ensuite, les règles de classification peuvent être construites. On considère la règle R_{ij}^{IJ} comme étant la règle correspondant au sous–espace flou [88, 53] $A_i^I \times B_j^J$. Elle peut être formulée comme dans (2.15), où C_{ij} est une des $|C|$ classes pré–définies.

Si x_1 est A_i^I et x_2 est B_j^J **alors** X appartient à la classe C_{ij} avec le degré de certitude CF_{ij} (2.15)

La procédure de génération des règles consiste à déterminer pour chaque couple (i, j) la classe C_{ij} à utiliser en conclusion de règle à partir des exemples d'apprentissage. Si l'on considère m points d'apprentissage $X_p = (x_{p1}, x_{p2}), p = \overline{1, m}$, la génération des règles est réalisée selon la procédure 2.2.

Début
  Pour chaque couple (i, j) faire /*pour chaque règle*/
    pour t = 1 à $|C|$ faire /*pour chaque classe*/
      $\beta_{C_t} = \sum_{x_p \in C_t} a_i^I(x_{p1}) \times b_j^J(x_{p2})$
    finpour
    Trouver la classe x pour laquelle $\beta_{C_x} = \max \beta_{C_1}, \ldots, \beta_{C_{|C|}}$.
    $C_{ij} = x$
    $CF_{ij} = \dfrac{\beta_{C_x} - \sum_{t=1, t \neq x}^{|C|} \frac{\beta_{C_t}}{|C| - 1}}{\sum_{t=1}^{|C|} \beta_{C_t}}$
  Finpour
Fin

Procédure 2.2 – Génération des règles floues de classification

Avec l'ensemble des règles $S = \{R_{ij}^{IJ}, i = \overline{1, I}, j = \overline{1, J}\}$ ainsi généré, on peut classifier une nouvelle entrée (x_1, x_2) [69] selon la procédure donnée dans 2.3. Si la classe x ne peut être déterminée de façon unique, l'entrée (x_1, x_2) est considérée inclassifiable ou une autre technique de "défuzzification" est utilisée (ligne 2).

De manière générale, les règles floues utilisées ici sont dites conjonctives dans la mesure où le "si ... alors ..." est interprété comme un "et". Dans la procédure 2.3, ligne 1, l'opérateur produit ($\times$) est utilisé comme opérateur de t–norme ("et" flou) à la fois pour implémenter le "et" présent dans la prémisse de la règle (2.15) et le "si ... alors ...". Les contributions des différentes règles sont ensuite agrégées disjonctivement par l'opérateur de t–conorme max (lignes 1 et 2).

Début
  Pour t = 1 à $|C|$ faire
    /*Calcul de l'activation de l'ensemble de règles S pour la classe t*/
    $\alpha_{C_t} = \max\{a_i^I(x_1) \times b_j^J(x_2) \times CF_{ij}; t = C_{ij}\}$ /*ligne 1*/
  Finpour
  Trouver la classe x pour laquelle $\alpha_{C_x} = \max \alpha_{C_1}, \ldots, \alpha_{C_{|C|}}$ /*ligne 2*/
Fin

Procédure 2.3 – Utilisation des règles floues de classification

Une extension des procédures présentées consiste à découper uniquement les zones de la grille considérées comme "intéressantes" ou "difficiles". La grille peut ainsi être raffinée avec des granularités différentes, qui théoriquement peuvent devenir infiniment petites, comme montré sur la figure 2.8 dans le cas 2D. En pratique, compte tenu des limitations en termes de pouvoir de calcul et de mémoire, la granularité est limitée. Une granularité très petite correspond à un nombre important de règles à stocker et à évaluer.

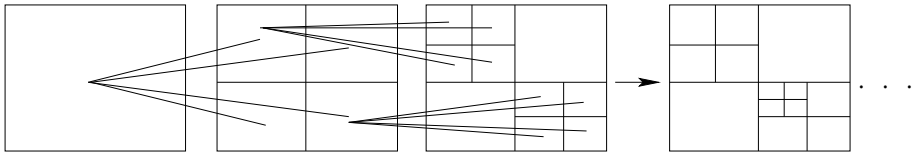


FIGURE 2.8 – Raffinage successif de la grille en augmentant le nombre de règles

Les procédures 2.2 et 2.3 peuvent être facilement généralisées au cas n–dimensionnel. Si on considère un objet X caractérisé par les attributs $x_k, k > 2$, qui doit être alloué à une des classes, une règle floue générale a alors la forme [74] :

$$\begin{array}{c} R_i : \textbf{Si } x_1 \textit{ est } A_{i1} \textit{ et } \ldots \textit{ et } x_k \textit{ est } A_{ik} \textit{ et } \ldots \textit{ et } x_n \textit{ est } A_{in}, \\ \textbf{alors } X \textit{ appartient à la classe } C_i \textit{ avec le degré de certitude } CF_i \end{array} \tag{2.16}$$

Dans l'équation (2.16), A_{ik} représente le sous–ensemble flou associé à l'attribut k dans la règle i.

Un aspect important des systèmes de classification basés sur des règles floues est le fait que les règles peuvent être exprimées dans un langage naturel [117]. Ainsi, leur interprétabilité par un expert est assez immédiate et l'interaction avec l'utilisateur humain est facilitée.

D'autres algorithmes d'apprentissage de règles floues de classification ont été développés, comme par exemple dans [135, 142] ou bien [128]. Des problèmes plus spécifiques ont également été analysés, comme par exemple la classification des données groupées dans des classes de forme non–convexe [36] ou l'utilisation de fonctions d'appartenance spécifiques afin de définir des régions ovales dans l'espace des attributs [1].

2.4.2 Règles d'association dans le domaine de la classification

D'autres types de règles, dites règles d'association, ont été utilisés en classification. Ces règles, provenant du domaine de la fouille de données [105, 2], sont basées sur le principe d'une recherche exhaustive de régularités dans les données : on essaie de trouver toutes les règles qui respectent des conditions imposées de fréquence et de confiance [80].

La construction des classifieurs basés sur les règles d'association se fait en deux étapes : la génération des règles, basée sur l'algorithme Apriori [2] et la construction du classifieur basé sur les règles ainsi déduites.

La génération des règles se fait elle aussi en plusieurs étapes. D'abord, on choisit toutes les entités de type "règle" qui ont un support plus grand qu'un seuil (le principe de support mimimum). Une telle entité est définie par la paire <*ensemble de conditions*, C_i>, où C_i est une étiquette de classe. La règle qui correspond à cette entité est donnée par :

$$\begin{array}{c} R_i : \textbf{Si } \textit{ensemble de conditions}, \\ \textbf{alors } X \textit{ appartient à la classe } C_i \end{array} \tag{2.17}$$

Le degré de confiance en R_i est donné dans ce cas par le rapport (2.18).

$$CF_i = \frac{\text{nombre d'exemples de la classe } C_i \text{ qui respectent l'ensemble de conditions}}{\text{nombre d'exemples d'apprentissage qui respectent l'ensemble de conditions}} \tag{2.18}$$

La deuxième étape consiste à comparer les entités ainsi obtenues et à en éliminer une partie selon le principe suivant : si deux entités ont le même ensemble de conditions seule celle ayant le degré de confiance maximal est gardée. Les règles qui survivent à cette étape sont nommées "règles possibles". Enfin, la dernière étape consiste à comparer les degrés de confiance des règles possibles ainsi obtenues avec un seuil pré–défini et à ne garder que celles qui ont un degré supérieur au seuil.

Une fois les règles obtenues, le classifieur est construit selon la procédure suivante : on ordonne l'ensemble de règles selon leur degré de confiance (ordre décroissant). Pour chaque classe on aura donc un jeu de règles ordonné, chacune des règles le composant ayant son propre degré de confiance. Les jeux de règles sont ensuite testés individuellement sur les données d'apprentissage. Si l'utilisation d'une règle n'améliore pas la précision de la classification, son degré de confiance est considéré comme insuffisant. Elle est donc éliminée du jeu de règles, ainsi que les règles qui la suivent (qui ont des degrés de confiance encore plus petits).

Différentes variantes de ce type de règles sont disponibles dans la littérature, comme dans [83], où les règles d'un jeu de règles sont ordonnées selon l'intensité de l'implication des règles à la place de leur degré de confiance, ou bien dans [102], où le nombre de règles est limité dès la phase de leur construction tout en gardant toutes les règles qui apportent de l'information utile.

Le règles d'association ont été aussi utilisées dans des systèmes "hybrides", comme par exemple en [113], où les règles d'association sont utilisées en conjonction avec les principes de la classification de Bayes afin de pouvoir analyser des ensembles de données de très grande taille.

2.4.3 Règles floues graduelles

Le concept de "règle graduelle" est apparu assez récemment, dans les 20 dernières années. Les principaux ouvrages dans cette direction reviennent à H. Prade et D. Dubois [42, 43]. Depuis leur apparition dans les années 1990 [41], les règles floues graduelles ont été principalement utilisées dans le domaine du traitement de l'imprécision [54, 146] avec quelques rares applications en contrôle des systèmes [6, 40]. En analysant leur principe, une autre application peut être imaginée : leur utilisation comme règles de classification.

Afin d'introduire les règles graduelles dans le contexte général des règles floues, les notions d'"information négative" et "information positive" sont nécessaires. Considèrons une affirmation a à laquelle on associe une notion de vérité "$Vrai$" ou "$Faux$". Cette affirmation contient de l'information négative si toute situation où a est fausse est impossible, sans garantir que toute situation où elle est vraie est possible. Ce type d'information est donné en spécifiant un interdit. Par contre, l'affirmation a représente de l'information positive si toute situation où a est vraie est possible, sans garantir que toute situation où elle est fausse est impossible. L'information positive est donnée empiriquement, à l'aide d'exemples observés [12]. Autrement dit, l'information négative impose des contraintes qui éliminent des situations impossibles, alors que l'information positive formalise des situations possibles [43].

Si l'on revient à une modélisation par règles, il apparaît que l'information positive correspond à une règle conjonctive alors que l'information négative correspond à une règle implicative où le "si ... alors ..." est interprété par une implication au sens logique du terme. La relation entrée/sortie associée à une collection de règles conjonctives est obtenue par disjonction des relations élémentaires. Au contraire dans le cas de règles implicatives une conjonction des relations élémentaires est réalisée, traduisant le fait que toutes les contraintes doivent être satisfaites.

Les règles implicatives sont exprimées en langage naturel sous la forme (2.19), ou encore en langage plus formel sous la forme (2.20) [43], où "$\longrightarrow$" est un opérateur d'implication. Dans (2.19), X

représente la variable d'entrée et Y la variable de sortie, sur laquelle on veut obtenir de l'information [42].

$$R_i : \textbf{Si } X \textit{ est } A_i \textbf{ alors } Y \textit{ doit être } B_i \tag{2.19}$$

$$R_i : A_i \longrightarrow B_i \tag{2.20}$$

Selon la classe de l'implication floue utilisée pour modéliser les règles implicatives, on aboutit à deux sous–familles de règles implicatives, appelées respectivement règles de certitude et règles graduelles. Les règles de certitude reposent sur l'utilisation de S–implications alors que les règles graduelles exploitent des R–implications. Dans ce contexte, une traduction plus précise de (2.19) devient (2.21).

$$\begin{array}{l} R_i : \textbf{Plus } X \textit{ est } A_i, \textbf{ plus} \text{ il est certain que } Y \textit{ est } B_i \text{ dans le cas d'une règle de certitude, et} \\ R_i : \textbf{Plus } X \textit{ est } A_i, \textbf{ plus } Y \textit{ est } B_i \text{ dans le cas d'une règle graduelle} \end{array} \tag{2.21}$$

En logique floue, plusieurs fonctions peuvent être associées à l'opérateur d'implication, comme montré dans l'équation (2.22), où $a, b \in [0, 1]$ représentent des degrés de vérité.

$$\begin{array}{ll} \text{Implication de Gődel :} & a \longrightarrow b = \begin{cases} 1; & a \leq b \\ b; & a > b \end{cases} \\ \text{Implication de Rescher–Gaines :} & a \longrightarrow b = \begin{cases} 1; & a \leq b \\ 0; & a > b \end{cases} \\ \\ \text{Implication de Lukasiewicz :} & a \longrightarrow b = min(1 - a + b, 1) \\ \text{Implication de Kleene–Dienes :} & a \longrightarrow b = max(1 - a, b) \end{array} \tag{2.22}$$

L'équation (2.22) regroupe les implications floues les plus couramment utilisées. Ces implications floues ne sont pas construites de façon désordonnée. Selon la façon dont elles généralisent l'implication de la logique classique elle se répartissent en deux grandes classes d'implications, appelées respectivement R–implications (ou implications résiduées) et S–implications. Les R–implications sont construites en généralisant le théorème de la déduction au cas flou selon le principe que ce qui est déduit est toujours au moins aussi vrai que ce qui a permis de faire la déduction. Quant aux S–implications, elles sont obtenues en généralisant la formule de la logique classique $p \longrightarrow q \equiv \rceil p \vee q$, où $\rceil$ et $\vee$ sont les opérateurs de négation et de disjonction. Les implications de Gődel et Rescher–Gaines sont des R–implications, celle de Kleene–Dienes une S–implication et celle de Lukasiewicz à la fois une R et une S–implication [84].

En conclusion, les règles graduelles représentent des contraintes dont chacune apporte de l'information négative [43]. Ainsi elles sont complémentaires aux règles conjonctives, qui apportent de l'information positive. L'ajout d'une règle graduelle équivaut à ajouter des zones "interdites", alors que l'ajout d'une règle conjonctive équivaut à ajouter des zones "permises". Autrement dit, les règles conjonctives désignent des régions de l'espace auxquelles l'entité étudiée peut appartenir, alors que les règles graduelles interdisent des régions de l'espace. En conséquence, l'utilisation de règles graduelles peut amener à des situations incohérentes, où l'on interdit tout l'espace de représentation. Par exemple, deux règles graduelles du type "***Si*** *X est A* ***alors*** *Y est B*" et "***Si*** *X est A* ***alors*** *Y est non–B*" excluent chacune l'espace permis par l'autre, car l'interprétation implicative de ces règles

est "***Si*** *X est A* ***alors*** *Y doit être B*" et respectivement "***Si*** *X est A* ***alors*** *Y doit être non–B*" [43]. Par contre, les deux mêmes règles considérées conjonctives, permettent au vecteur Y de prendre des valeurs dans tout l'espace de sortie, car elles sont interpretées comme "***Si*** *X est A* ***alors*** *Y peut être B*" et respectivement "***Si*** *X est A* ***alors*** *Y peut être non–B*".

Le chapitre suivant, qui présente le principe de base du système de classification proposé dans cette thèse, explique plus en détails le principe de fonctionnement des règles graduelles, ainsi que quelques propriétés qui facilitent leur apprentissage et les rendent très efficaces dans le processus de classification.

2.4.4 Modèles graduels

Avant introduire le principe de classification à base de règles floues graduelles, développé dans la thèse, une brève présentation des modèles graduels termine ce chapitre dédié à l'état de l'art. Les modèles graduels sont construits à partir d'un ensemble de motifs graduels extraits à partir d'un ensemble d'apprentissage par des techniques de fouille de données. Il est clair qu'un lien étroit existe entre les notions de règles graduelles et de motifs graduels, bien que ce point n'ait pas été approfondi dans le cadre de cette thèse.

Des systèmes de classification basés sur ces modèles ont été proposés, par exemple dans [23]. Ces systèmes arrivent à classifier des données dans le cas où les valeurs d'attributs considérées individuellement ne permettent pas une individualisation des différentes classes. Le principe de base de l'approche est d'analyser la tendance de certains sous–ensembles d'attributs plutôt que les attributs pris séparément. Ainsi on prend en considération une possible corrélation entre la tendance d'un attribut i à être "plutôt petit" quand un autre attribut j est "plutôt grand" si l'exemple appartient à une classe donnée, alors que les deux attributs ne sont pas corrélés si l'exemple n'appartient pas à la classe.

Par exemple, dans le domaine de la génétique pour les cas de tumeurs malines, des liens du type "plus le gène G_1 est exprimé, moins le gène G_2 est exprimé" sont mis en évidence [23].

La classification basée sur des modèles graduels est faite en deux étapes : 1. la construction des modèles graduels qui définissent chaque classe et 2. la définition de la classe d'appartenance des nouveaux exemples à partir de ces modèles.

Les modèles graduels sont constitués des sous–ensembles d'attributs de l'ensemble d'attributs d'origine auxquels on associe un ordre qui prend en considération la tendance d'augmentation ou de diminution respective des attributs concernés. Quelques notions de base ont été définis dans [85] :

- les entités graduelles : chaque attribut est associé à un opérateur de comparaison qui donne sa tendance plutôt ascendante ou plutôt descendante pour les exemples d'une même classe. On note une telle entité (x_i, θ_i), où x_i est un attribut et θ_i prend une valeur dans l'ensemble $\{\leq, \geq\}$.
- les ensembles d'entités graduelles : différentes combinaisons des attributs individuels sont pris en considération. Un ensemble d'entités graduelles est ainsi défini comme $g = \{(x_1, \theta_1), \cdots (x_k, \theta_k)\}$. Une cardinalité minimale de 2 est nécessaire pour chaque ensemble.
- la cardinalité d'un ensemble d'entités graduelles : le nombre d'exemples dans l'ensemble d'apprentissage qui respecte la relation d'ordre définie par l'ensemble d'entités graduelles. Si on note $x_{i,p}$ l'attribut "i" de l'exemple d'apprentissage p, la cardinalité est donnée par le nombre d'exemples d'apprentissage qui respectent la relation $x_{i,p}\theta_i x_{i,p+1}$. On note cette cardinalité $\lambda(g)$.
- le support de l'ensemble des entités graduelles : le rapport entre la cardinalité $\lambda(g)$ et la cardinalité de l'ensemble d'apprentissage : $supp(g) = \frac{\lambda(g)}{|ensemble\ d'apprentissage|}$

– la discrimination d'un ensemble d'entités graduelles pour une classe C :

$$disc(g, C) = \frac{supp(g, C)}{\sum_{C_i \in C, supp(g,C_i) > minsupp} supp(g, C_i)}$$

Afin de calculer la cardinalité $\lambda(g)$, [85] propose d'utiliser la théorie des graphes : chaque exemple d'apprentissage est un nœud d'un graphe et deux nœuds sont liés si la relation d'ordre décrite par g est respectée.

A l'aide de ces notions, le classifieur peut être ensuite construit en deux étapes :

– calculer les ensembles d'entités graduelles les plus fréquents pour chaque classe (qui ont un support supérieur à un seuil pré–défini $minsupp$)
– pour chaque classe définir les règles de classification selon ces ensembles

Afin de classifier de nouveaux exemples deux situations sont considérées :

– plusieurs exemples à classifier sont disponibles et connus comme appartenant à la même classe : on calcule les ensembles d'entités graduelles les plus fréquents pour l'ensemble donné, en considérant le même seuil $minsupp$ et on les compare avec les ensembles déjà extraits pour les classes définies à partir de l'ensemble d'apprentissage. On note l'ensemble des nouveaux ensembles d'entités graduelles G_{new}. La classe des nouveaux exemples sera alors :

$$C_{new} = argmax_{C_i \in C}(\sum_{g \in G_{new}} disc(g, C_i)),$$

où $disc(g, C_i) = 0$ si g ne fait pas partie des ensembles calculés à partir de l'ensemble d'apprentissage.
– on dispose d'un seul exemple à classifier : pour chaque classe apprise et pour chaque ensemble g on détermine dans quelle mesure le nouvel exemple respecte la relation d'ordre définie par g. Le support de chaque ensemble est recalculé en rajoutant le nouvel exemple à l'ensemble d'apprentissage. Pour chaque classe C_i on calcule une "pertinence", qui est la somme des discriminations obtenues pour chaque ensemble d'entités graduelles g associés à cette classe pour l'ensemble d'apprentisage augmenté. Enfin, l'exemple est placé dans la classe pour laquelle cette pertinence est la plus grande.

Chapitre 3

Classification basée sur des règles graduelles - principe

3.1 Approche de base

Le système proposé se place dans le cadre de la classification floue à base de règles. L'approche usuelle de la classification à base de règles est d'utiliser des règles "**Si** ... **alors**", comme expliqué dans le chapitre précédent. Les règles graduelles s'inscrivent dans le contexte général de ce type de classification, mais elles introduisent des changements d'interprétation des règles et de la méthode d'inférence à utiliser. L'objectif de cette section est d'illustrer le principe de la méthode proposée. Pour cela, on se situe dans un espace formé de deux attributs x_1 et x_2 et on se limite à la représentation d'une seule classe notée C_1.

La forme typique d'une règle graduelle est :

$$\textbf{Plus } x_1 \textit{ est } F_1, \textbf{ Plus } x_2 \textit{ est } F_2, \tag{3.1}$$

avec F_1 et F_2, deux fonctions d'appartenance. Cette forme est en fait une variante particulière des règles du type :

$$\textbf{Si } x_1 \textit{ est } F_1, \textbf{ alors } x_2 \textit{ est } F_2 \tag{3.2}$$

Une règle graduelle de la forme décrite par l'équation (3.1) sera évaluée à l'aide d'une implication. Une implication est une fonction notée "$a \longrightarrow b$" qui associe à tout couple (a, b) de $[0, 1] \times [0, 1]$ un degré dans l'intervalle [0, 1]. Pour des règles graduelles, seules des implications résiduées sont utilisables [13].

Trois exemples d'implications résiduées ont été présentés dans le chapitre précédent (l'implication de *Gödel*, l'implication de $Rescher - Gaines$ et l'implication de *Lukasiewicz*). Dans le cadre de cette thèse, l'implication de $Rescher - Gaines$, rappelée dans (3.3), a été considérée pour l'ensemble des implémentations et tests.

$$a \longrightarrow b = \begin{cases} 1 & ; \quad a \leq b \\ 0 & ; \quad a > b \end{cases} \tag{3.3}$$

D'un point de vue général, une règle graduelle du type (3.1) peut être associée à un graphe Γ qui

est défini par :

$$\Gamma(x_1, x_2) = \mu_{F_1}(x_1) \longrightarrow \mu_{F_2}(x_2) \tag{3.4}$$

Afin d'intégrer ce type de règles dans un système de classification, les règles doivent être modifiées. La solution choisie est de placer chaque attribut analysé dans la partie "prémisse" de la règle, alors que la partie "conclusion" doit contenir une information liée aux classes recherchées par le système. Pour conserver la sémantique proposée, la relation d'implication entre les attributs doit être gardée et transferée dans la partie prémisse de la règle. Une règle définie selon le formalisme (3.5) répond efficacement à tous ces besoins [29]. Si l'on transforme la règle (3.5) en utilisant directement l'implication dans la partie prémisse, on obtient la forme finale proposée afin d'exprimer la règle, c'est–à–dire l'expression formulée en (3.6).

$$\begin{array}{c}\textbf{Si } (\textbf{Plus } x_1 \textit{ est } F_1, \ \textbf{Plus } x_2 \textit{ est } F_2) \\ \textbf{alors } \textit{classe } C_1\end{array} \tag{3.5}$$

$$\begin{array}{c}\textbf{Si } x_1 \textit{ est } F_1, \longrightarrow x_2 \textit{ est } F_2 \\ \textbf{alors } \textit{classe } C_1\end{array} \tag{3.6}$$

La figure 3.1 illustre le fonctionnement d'une telle règle de classification pour des fonctions d'appartenance linéaires. A partir de cette figure, quelques observations sont nécessaires pour expliquer la démarche. Le système reçoit comme entrée le vecteur d'attributs $X = \{x_1, x_2\}$. Il applique à chacun d'entre eux la fonction d'appartenance qui lui correspond. Jusqu'alors on considére ces fonctions d'appartenance connues et fournies par un système d'apprentissage des règles qui sera détaillé ultérieurement. Les deux fonctions $\mu_{F_1}(x_1)$ et $\mu_{F_2}(x_2)$ sont exprimées analytiquement par les équations (3.7) et (3.8).

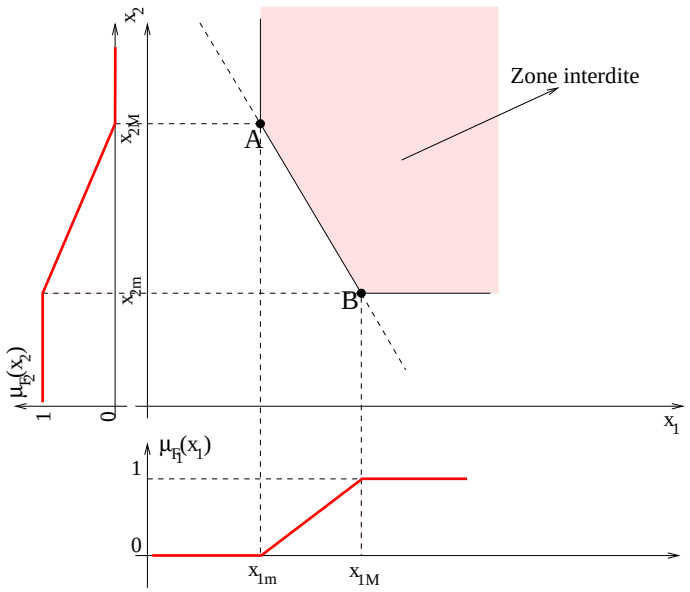


FIGURE 3.1 – Fonctions d'appartenance et graphe résultant

$$\mu_{F_1}(x_1) = \begin{cases} 0 & ; \quad x_1 < x_{1m} \\ \frac{x_1}{x_{1M} - x_{1m}} + \frac{x_{1m}}{x_{1m} - x_{1M}} & ; \quad x_{1m} \leq x_1 \leq x_{1M} \\ 1 & ; \quad x_{1M} < x_1 \end{cases} \tag{3.7}$$

$$\mu_{F_2}(x_2) = \begin{cases} 1 & ; \quad x_2 < x_{2m} \\ \frac{x_2}{x_{2m}-x_{2M}} + \frac{x_{2M}}{x_{2M}-x_{2m}} & ; \quad x_{2m} \leq x_2 \leq x_{2M} \\ 0 & ; \quad x_{2M} < x_2 \end{cases} \tag{3.8}$$

Les fonctions d'appartenance permettent de "projeter" chaque attribut dans l'intervalle [0, 1]. Une fois les valeurs des degrés d'appartenance calculées, le système utilise celles-ci comme opérandes de l'opérateur d'implication. Le résultat de cette dernière étape, basée sur l'implication de Rescher-Gaines, est une valeur dans l'ensemble $\{0, 1\}$, qui, dans ce cas, signifie l'appartenance ou la non-appartenance à la classe considérée. Le résultat final peut être visualisé dans la figure 3.1 et consiste à partager le plan des deux attributs en deux zones : une zone "permise" qui, conformément à la règle décrite, correspond aux points qui peuvent appartenir à la classe et une zone "interdite" qui est composée par des points qui n'appartiennent pas à cette classe.

Jusqu'ici, la règle permet d'allouer une zone infinie à la classe. Si l'on considère une classe définie par un nuage de points, un système de classification basé sur des règles doit en principe trouver une manière plus précise de définir les frontières (donc les critères d'identification) de cette classe. Pour y arriver, une simple observation s'impose sur le système décrit : le rajout d'une règle du type (3.1) et l'application d'un opérateur de conjonction entre les graphes des deux règles revient dans l'espace des attributs à rajouter une zone interdite. Du point de vue de la formalisation, cela revient à introduire de nouvelles contraintes dans la partie prémisse de la règle. Une telle règle, avec n contraintes, va réduire l'espace alloué à la classe à une forme plus précise, donc mieux adaptée à la forme de la classe. Une règle formée de plusieurs contraintes a la forme générale donnée par l'équation (3.9). Les contraintes doivent être cohérentes, c'est-à-dire qu'elle doivent définir des surfaces permises qui ont une intersection non vide. A partir de trois contraintes, la surface définie peut représenter une surface fermée, qui ne peut être qu'un polygone convexe.

$$\begin{array}{c} \textbf{Si} \\ x_1 \; est \; F_{0,1} \longrightarrow x_2 \; est \; F_{0,2} \; \textbf{et} \\ x_1 \; est \; F_{1,1} \longrightarrow x_2 \; est \; F_{1,2} \; \textbf{et} \\ \vdots \\ x_1 \; est \; F_{n-1,1} \longrightarrow x_2 \; est \; F_{n-1,2} \\ \textbf{alors} \; classe \; C_1 \end{array} \tag{3.9}$$

Dans l'équation (3.9), F_{ji} signifie : le symbole j appliqué sur l'attribut i, où $i \in 1, 2$. D'un point de vue mathématique, la règle (3.9) est formulée par l'expression (3.10), où T représente une t–norme, c'est–à–dire un "et" flou.

$$\mu_{C_1}(x_1, x_2) = \Gamma(x_1, x_2) = \top\Big(\mu_{F_{i1}}(x_1) \longrightarrow \mu_{F_{i2}}(x_2)\Big), \; \forall i = \overline{1, n} \tag{3.10}$$

Quelques problèmes n'ont encore pas été détaillés. Le premier porte sur le choix des fonctions d'appartenance les plus pertinentes. Le deuxième, étroitement lié, est en fait un problème de représentation des contraintes d'un point de vue numérique. Le découpage de l'espace des attributs en les deux sous-espaces, "interdit" et "permis", tel qu'il a été présenté dans la figure 3.1, est basé implicitement sur l'hypothèse d'espace fini dans la mesure où les valeurs limites x_{1m}, (resp. x_{1M}) et x_{2m}, (resp. x_{2M}) sont des grandeurs finies. Or, la partie "intéressante" des fonctions d'appartenance réside seulement dans leur partie non-constante, où la pente est différente de 0. Il en résulte que le choix d'intervalles bornés pour $[x_{1m}, x_{1M}]$ et $[x_{2m}, x_{2M}]$ ne permet de définir un séparateur linéaire que sur un segment ($[AB]$ dans le cas de la figure 3.1). Une règle qui délimiterait deux demi-plans

dans tout l'espace 2D considéré, évidemment infini, nécessiterait que les points A et B soient repoussés à l'infini sur la droite $[AB]$. Une telle définition de règles est impossible d'un point de vue représentation et calcul, mais aussi d'un point de vue conceptuel (interprétation de la signification floue).

De manière intuitive mais également applicative, les contraintes linéaires associées aux classes seront dorénavant définies dans un espace fini que l'on appelera "espace de travail". Pour chaque classe, il faut trouver les principes et les moyens permettant de délimiter cet "espace de travail". Cet aspect est developpé dans le paragraphe suivant.

3.2 Formes des classes et espace de travail

La méthode proposée est basée sur la possibilité de définir dans l'espace de travail 2D associé à une classe (le cadre dans lequel on se situe pour le moment) une surface qui peut être associée à cette classe. Pour montrer les capacités de représentation des règles proposées, on analyse dans un premier temps comment les zones interdites par une seules contrainte varient selon les fonctions d'appartenance de F_1 et F_2. On peut alors facilement remarquer qu'en inversant les signes des pentes des fonctions d'appartenance associées aux deux attributs, quatre cas, présentés dans la figure 3.2, peuvent être distingués.

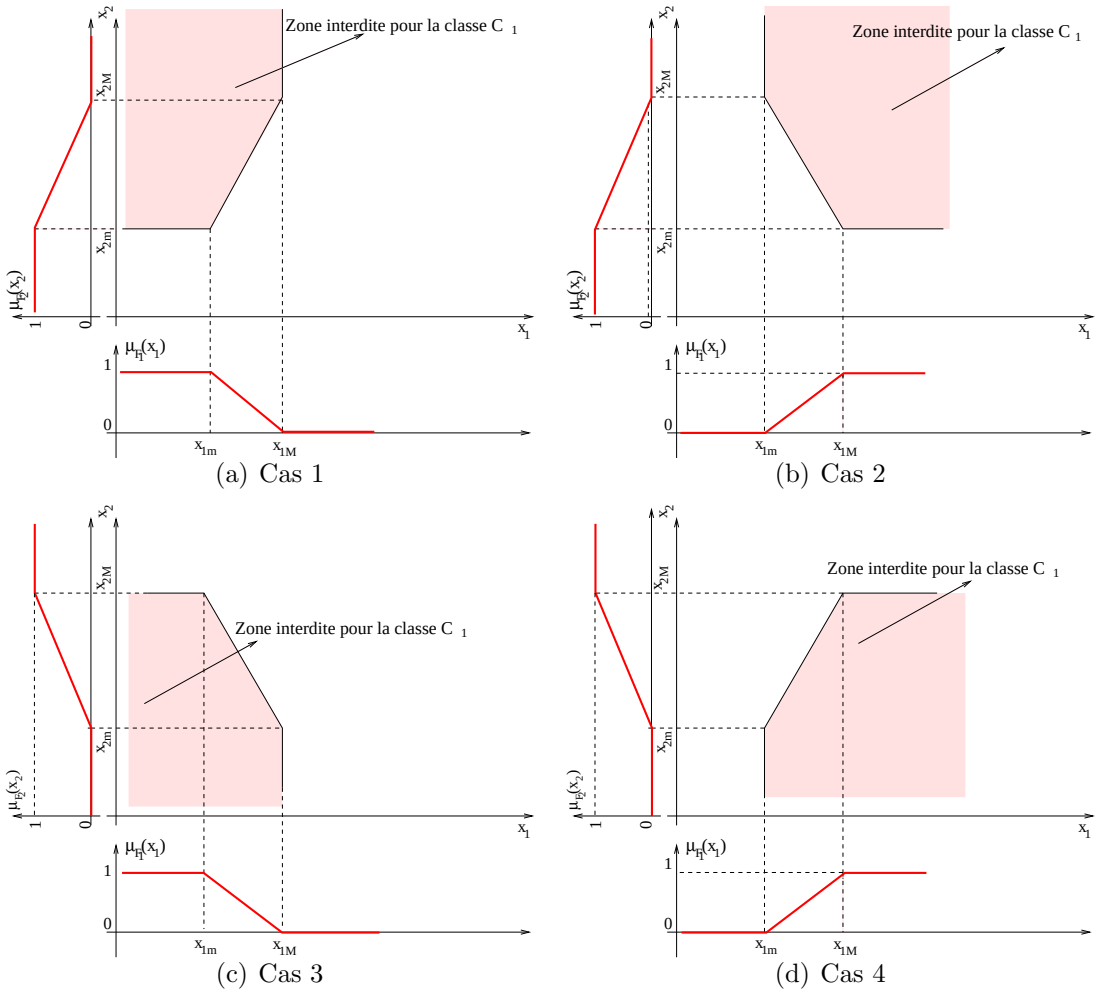


FIGURE 3.2 – Types de zones interdites et les fonctions d'appartenance qui les produisent

En analysant les fonctions d'appartenance présentées et les zones "interdites" et "permises" qui leur correspondent, une première conclusion s'impose : on peut "découper" dans l'espace des attributs une zone "permise", finie ou infinie, par simple agrégation conjonctive de plusieurs relations implicatives entre les deux attributs. Le résultat est en fait une règle composée de plusieurs contraintes, comme proposé dans l'équation (3.9). Si la zone "permise" par une telle règle est finie, elle prend la forme d'un polygone convexe. On se limitera donc, dans le cadre de cette étude, à la représentation des polygones convexes. L'obtention de ce polygone pour une classe recherchée sera analysée ultérieurement. Dans les cas où la forme géométrique ne serait pas convexe, il est toujours possible de se ramener à un ensemble de polygones convexes.

Les frontières de la classe analysée sont donc établies comme les côtés d'un polygone convexe supposé connu. Il faut maintenant trouver la manière de construire la règle de classification composée de plusieurs contraintes qui délimite dans l'espace de travail une zone "permise" identique à ce polygone convexe. Comme déjà anticipé dans le paragraphe précédent, il faut d'abord limiter l'espace de travail à un espace fini, tout en préservant, pour des raisons de cohérence du système, le principe de base des règles graduelles. La limitation de cet espace doit être simple d'un point de vue formel et ne doit pas nécessiter de traitements supplémentaires importants.

Une approche simple consiste à limiter l'espace de travail à un rectangle parallèle aux axes du système de référence considéré qui englobe au plus près le polygone. C'est une approche assez intuitive qui possède quelques avantages assez importants, qui seront discutés par la suite. Pour illustrer le principe, on considère le cas du polygone $MNOPQRS$ représenté dans la figure 3.3(a). La figure 3.3(b) représente la restriction de l'espace original infini au rectangle représenté ($ABCD$). Afin de rester dans un même espace de représentation, il faut trouver la règle implicative qui délimite cet espace de travail. Un tel rectangle est assez particulier d'un point de vue formel dans le contexte des règles graduelles. Il est caractérisé par des pentes nulles et infinies, donc les fonctions d'appartenance qui doivent être déduites ont une forme différente des formes présentées dans le paragraphe précédent.

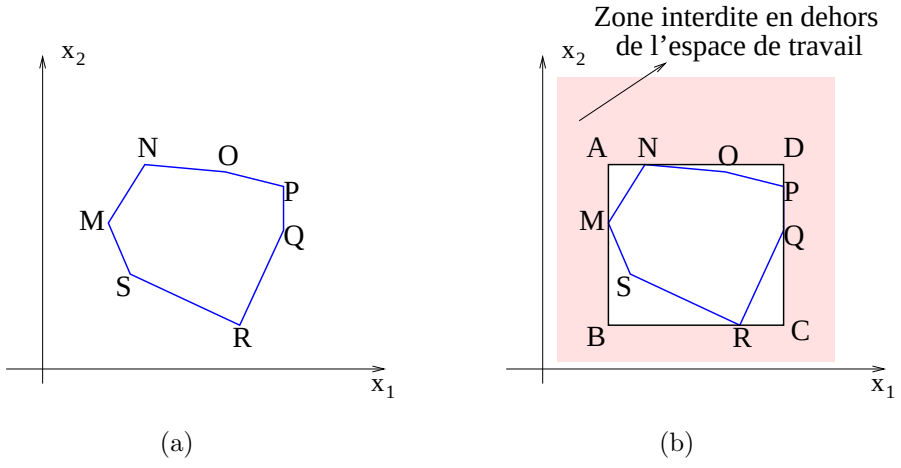


FIGURE 3.3 – La réduction de l'espace de travail à un espace fini

En tenant compte des définitions données par les équations (3.3) et (3.9), une solution pour interdire l'espace situé en dehors du rectangle indiqué est une règle composée de deux contraintes :

$$\begin{array}{c} \textbf{Si} \\ x_1 \; est \; F_{0,1} \longrightarrow x_2 \; est \; F_{0,2} \; \textbf{et} \\ x_1 \; est \; F_{1,1} \longrightarrow x_2 \; est \; F_{1,2} \\ \textbf{alors} \; classe \; C_1 \end{array} \tag{3.11}$$

En notant "$\mu_{F_{j,i}}(x_i)$" le degré de vérité de la proposition "x_i *est* $F_{j,i}$", la règle (3.11) s'écrit de manière équivalente sous la forme donnée par l'équation (3.12).

$$\begin{array}{c} \textbf{Si} \\ \mu_{F_{0,1}}(x_1) \longrightarrow \mu_{F_{0,2}}(x_2) \textbf{ et} \\ \mu_{F_{1,1}}(x_1) \longrightarrow \mu_{F_{1,2}}(x_2) \\ \textbf{alors } classe\ C_1, \end{array} \tag{3.12}$$

Dans l'équation (3.12) les fonctions d'appartenance $\mu_{F_{j,i}}(x_i)$, $i = \overline{1,2}$, $j = \overline{0,1}$ sont données dans la figure 3.4(a) et 3.4(b). Quelques remarques s'imposent sur les formes de ces fonctions d'appartenance :

1. La première contrainte permet de couper les zones verticales tel qu'illustré sur la figure 3.4(a). La fonction d'appartenance de $F_{0,2}$ est une fonction constante, nulle sur tout son domaine de définition. Elle ne dépend pas du rectangle à représenter et gardera donc une forme identique pour toutes les règles construites selon le principe décrit. D'un point de vue ensembliste, sa signification est un "ensemble vide" puisqu'aucun élément du domaine de définition n'appartient à l'ensemble représenté par la fonction d'appartenance identiquement nulle. D'un point de vue logique, la proposition x_2 *est* $F_{0,2}$ est interprétée comme "FAUX" indépendamment de x_2, ce qui se traduit dans la contrainte qui lui correspond par l'interdiction des valeurs de x_1 n'appartenant pas à l'intervalle $[x_{1m}, x_{1M}]$, c'est-à-dire telles que la proposition "x_1 *est* $F_{0,1}$" est au moins un peu vraie ($\mu_{F_{0,1}}(x_1) \neq 0$).
2. La fonction d'appartenance de $F_{1,1}$ est aussi une fonction constante, égale à 1 indépendamment du rectangle à définir. L'ensemble ainsi représenté est le domaine de définition de l'attribut x_1 et la proposition correspondante est interprétée comme "VRAI", indépendamment de la valeur de x_1. Son utilisation en premier opérande de l'implication permet d'interdire les valeurs de x_2 n'appartenant pas totalement à $F_{1,2}$, c'est-à-dire n'appartenant pas à l'intervalle $[x_{2m}, x_{2M}]$. La contrainte permet donc de découper horizontalement l'espace (voir figure 3.4(b)).
3. En conclusion, chaque règle de la forme 3.9 aura deux contraintes en charge de délimiter l'espace de travail rectangulaire associé à la classe. Ces dernières, numérotées 0 et 1, traduisent dans le formalisme flou implicatif adopté les contraintes binaires

$$x_1 \notin [x_{1m}, x_{1M}] \longrightarrow FAUX \ et$$
$$VRAI \longrightarrow x_2 \in [x_{2m}, x_{2M}],$$

ce qui se résume finalement en $x_1 \in [x_{1m}, x_{1M}]$ *et* $x_2 \in [x_{2m}, x_{2M}]$.

3.3 Détermination des fonctions d'appartenance

Il reste maintenant à délimiter dans l'espace fini représenté par le rectangle $ABCD$ le polygone cible qui définit la classe : le polygone $MNOPQRS$ de la figure 3.3(b). Pour interdire les zones extérieures au polygone considéré, il suffit de construire une contrainte pour chacun de ses côtés. Ces contraintes ont pour seul but de définir comme zone "interdite" la surface extérieure au polygone, située entre les côtés du polygone et les limites de l'espace de travail.

Une première solution consiste à prolonger chaque côté du polygone afin d'obtenir les intersections avec les limites de l'espace de travail. Deux remarques peuvent alors être formulées :

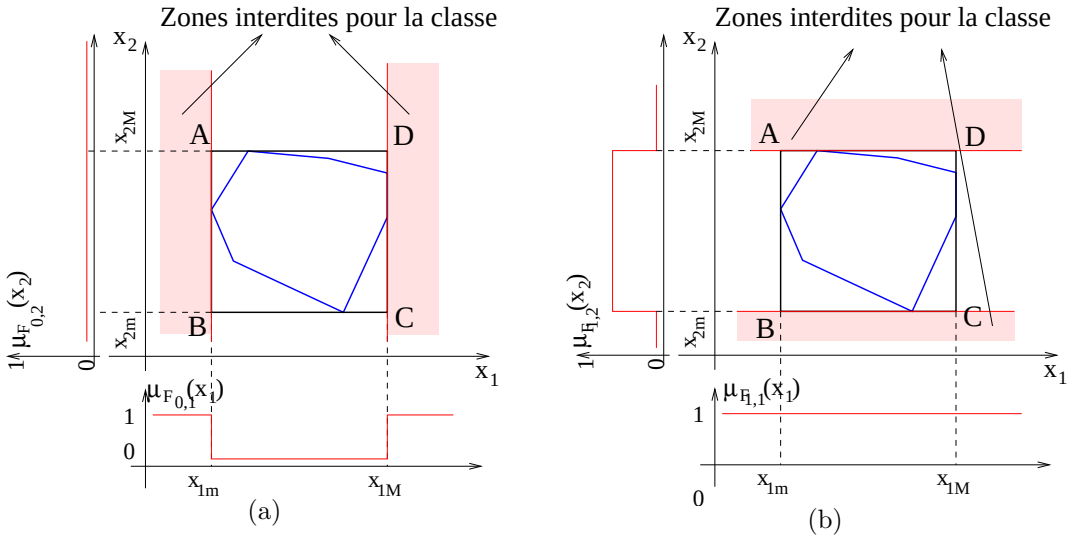


FIGURE 3.4 – Limitation de l'espace de travail réservé à la classe

1. Entre le rectangle englobant $ABCD$ et les droites–support des segments du polygone, il existe toujours au moins deux points d'intersection.
2. Comme le polygone est convexe, les droites–support des segments du polygone ne "coupent" pas le polygone.

A partir des deux points d'intersection obtenus, il est possible de déterminer les fonctions d'appartenance qui permettront d'obtenir les contraintes. La figure 3.5 présente les différentes étapes nécessaires dans la détermination de l'ensemble des contraintes.

Deux cas particuliers peuvent être évoqués lorsque les extrémités des segments du polygone se trouvent sur le rectangle englobant :

1. Les deux extrémités peuvent être sur le même côté du rectangle englobant. Dans ce cas, le segment est entièrement confondu avec la frontière de l'espace de travail. C'est le cas du segment $[PQ]$ sur la figure 3.5. Dans cette situation, aucun traitement supplémentaire n'est nécessaire pour ce côté, puisqu'il est déjà géré par les contraintes associées au rectangle englobant.
2. Les extrémités peuvent être sur des côtés différents du polygone englobant. C'est le cas du côté $[MN]$, qui a ses deux extrémités sur le rectangle englobant $ABCD$. Dans cette situation les deux points d'intersection entre le rectangle et la droite–support de $[MN]$ sont précisément les points M et N. Le même cas se produit pour le côté $[QR]$.

En dehors de ces deux situations, la prolongation de la droite–support est nécessaire de façon à déterminer les deux points d'intersection avec le rectangle $ABCD$, comme présenté dans les figures 3.5(a), 3.5(b), 3.5(d) et 3.5(e). Ces points d'intersection sont ensuite exploités pour construire les fonctions d'appartenance appropriées.

Pour la lisibilité des graphes présentés dans la figure 3.5 quelques précisions peuvent être utiles. Les zones interdites par les nouvelles fonctions d'appartenance sont colorées. Comme l'espace de travail est réduit au rectangle $ABCD$, les fonctions d'appartenance ont une signification et une importance pratique uniquement dans cet espace. Leur prolongation est quand même représentée en dehors de ces limites en ligne pointillée, pour des raisons de consistence. Un point dans l'espace des attributs peut prendre des valeurs dans tout le domaine, donc toutes les contraintes doivent être applicables sur tout le domaine. L'effet de ces parties des fonctions d'appartenance dans l'espace des

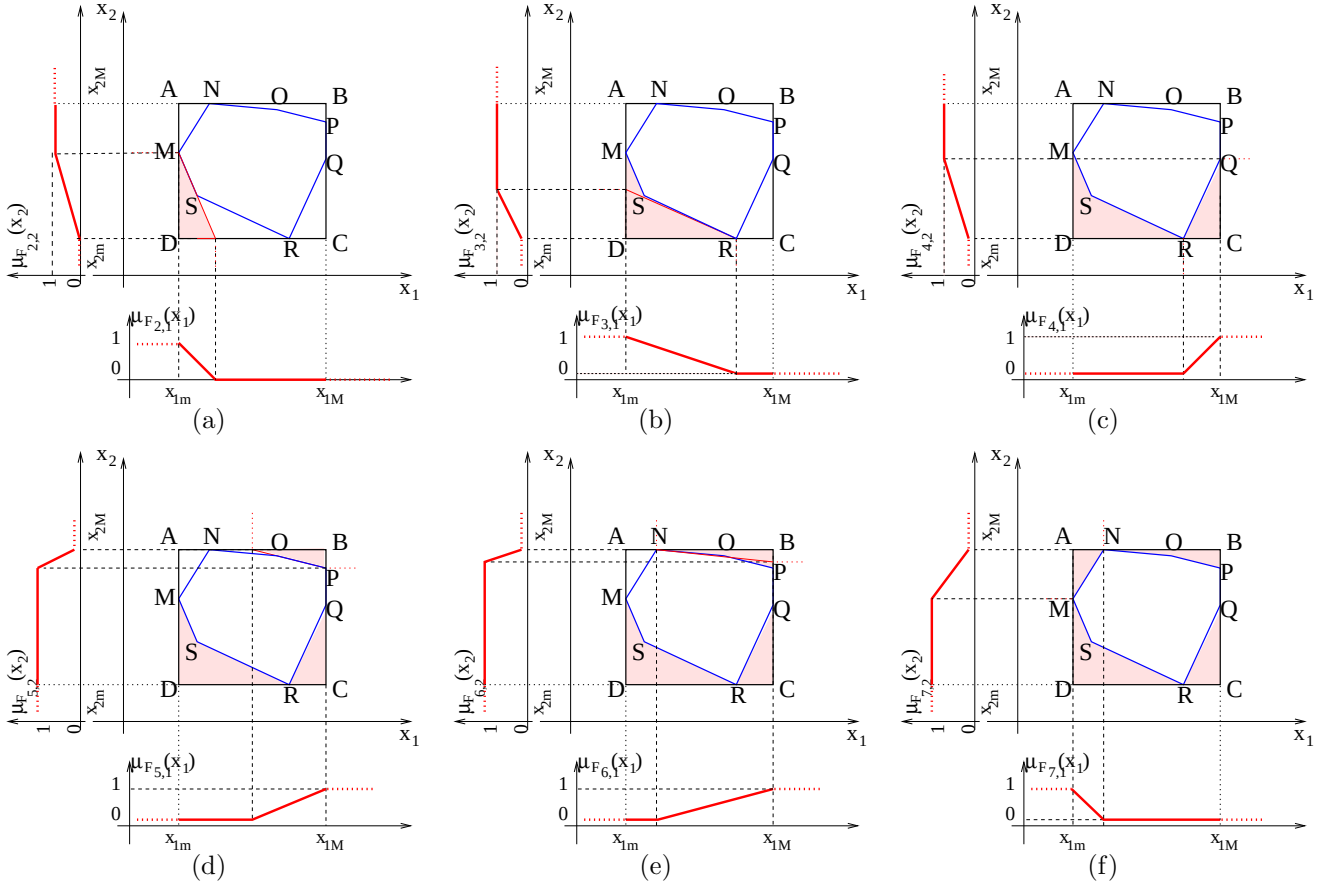


FIGURE 3.5 – Construction des fonctions d'appartenance – approche de base

attributs est aussi marqué avec des lignes pointillées dans l'extérieur du rectangle $ABCD$, mais pour des raisons de lisibilité l'espace qu'elles interdisent n'a pas été coloré.

Une fonction d'appartenance "typique", comme celles présentées dans la figure 3.5 sera définie par trois composantes :

- la valeur gauche : $F_{i,j}(x_j).g = x_{jm},\ j = \overline{1,2}\ i = \overline{2, N_C + 1}$
- la valeur droite : $F_{i,j}(x_j).d = x_{jM},\ j = \overline{1,2}\ i = \overline{2, N_C + 1}$
- le signe de la pente : $F_{i,j}(x_j).p$, défini comme :
  - 1 pour une fonction d'appartenance du type :

$$\mu_{F_{i,j}(x_j)} = \begin{cases} 0; & x_j < x_{jm} \\ \frac{x_j}{x_{jM}-x_{jm}} + \frac{x_{jm}}{x_{jm}-x_{jM}}; & x_{jm} \leq x_j \leq x_{jM} \\ 1; & x_{jM} < x_j \end{cases}$$

  - -1 pour une fonction du type :

$$\mu_{F_j(x_j)} = \begin{cases} 1; & x_j < x_{jm} \\ \frac{x_j}{x_{jm}-x_{jM}} + \frac{x_{jM}}{x_{jM}-x_{jm}}; & x_{jm} \leq x_j \leq x_{jM} \\ 0; & x_{jM} < x_j \end{cases}$$

Les étapes de l'algorithme sont tout d'abord décrites en langage naturel dans la procédure 3.1, puis l'algorithme détaillé correspondant est donné par l'algorithme 42.

Début
Etape 1. Vérifier que le segment traité n'est pas vertical ou horizontal
Etape 2. Trouver les intersections de la droite support du segment avec le rectangle englobant – points $H(h_1, h_2)$ et $J(j_1, j_2)$
Etape 3. Placer les points caractéristiques des fonctions d'appartenance en utilisant les deux points H et J
Etape 4. Choisir les pentes des fonctions d'appartenance en fonction du quadrant trigonométrique dans lequel se trouve le segment en considérant un système de référence centré sur H, le point d'origine du segment analysé $[HJ]$
Fin

Procédure 3.1 – Algorithme de base – langage naturel

L'algorithme 42 présente les traitements nécessaires pour obtenir les fonctions d'appartenance associées au polygone, en considérant l'existence des deux contraintes qui limitent l'espace de travail. L'obtention de ces deux contraintes est immédiate, comme présenté préalablement dans la figure 3.4.

La convention qui a été utilisée concerne l'ordre associé aux points du rectangle englobant $ABCD$. Cet ordre est celui proposé dans la figure 3.3(b), notamment le point $A = (a_1, a_2)$ qui est le coin en haut à gauche et le sens de parcours qui est le sens trigonométrique.

3.4 Amélioration de la construction des fonctions d'appartenance

Même si d'un point de vue méthodologique l'approche proposée dans la section précédente est pertinente, elle présente quelques désavantages. En effet le système dispose des points qui définissent le polygone associé à la classe pour construire les contraintes, en fait les sous–ensembles flous sous–jacents. La solution présentée n'utilise cependant pas directement ces points, mais calcule de nouveaux points (intersections des côtés de la forme convexe avec les frontières de l'espace de travail) pour "calibrer" les fonctions d'appartenance. Ceci ajoute des calculs numériques, donc un temps de calcul additionnel, qui peut devenir important dans le cadre d'un système complexe. On peut également remarquer une redondance importante introduite par la méthode utilisée pour construire les règles. En effet deux côtés voisins vont interdire des zones communes dans l'espace de travail.

Pour améliorer l'efficacité du système en terme de nombre de calculs réalisés et pour optimiser la description des zones "interdites" en limitant les redondances, les contraintes peuvent être obtenues directement à partir des points P_i du polygone.

Le processus d'obtention des fonctions d'appartenance est donné par l'algorithme 37 et est illustré par la figure 3.6. En fait, même si le cœur des deux algorithmes est identique, le deuxième est un peu plus simple, parce qu'il utilise directement les coordonnées des points qui forment le polygone. Par contre il demande de vérifier certains aspects, qui vont être détaillés ci–après afin d'être certain des frontières de l'espace qu'il dèlimite. L'algorithme utilisé est présenté en language naturel dans la procédure 3.2.

Tout le raisonnement à l'origine du système de classification proposé est fondé sur un principe de base : à une classe définie dans un espace à deux attributs, on peut associer un polygone convexe. Ce polygone convexe peut alors être représenté par une règle graduelle de classification telle que

Algorithm 1 Algorithme pour l'obtention des sous–ensembles flous (l'approche de base)

Entrée : le polygone P associé à la classe ; son rectangle englobant $ABCD$
Sortie : $\{F_{k,i}\}, \forall k = \overline{2, N_C + 1},\ \forall i \in \{1, 2\}$
Variables locales : i, k, $H = (h_1, h_2)$, $J = (j_1, j_2)$, c = l'ordre de la contrainte courante

$c \longleftarrow 1$
pour $k = 1$ **à** $N - 1$ //Pour chaque point du polygone **faire**
$*Etape\ 1*$
si $p_{(k)1} \neq p_{(k+1)1}$ **et** $p_{(k)2} \neq p_{(k+1)2}$ **alors**
$*Etape\ 2*$
$c \longleftarrow c + 1$
$H = (h_1, h_2) \longleftarrow$ point d'intersection de la demi–droite $[P_{(k+1)}P_{(k)}$ avec le rectangle $ABCD$
$J = (j_1, j_2) \longleftarrow$ point d'intersection de la demi–droite $[P_{(k)}P_{(k+1)}$ avec le rectangle $ABCD$
$*Etape\ 3*$
$F_{c,1}(x_1).g \longleftarrow \min(h_1, j_1)$; $F_{c,2}(x_2).g \longleftarrow \min(h_2, j_2)$
$F_{c,1}(x_1).d \longleftarrow \max(h_1, j_1)$; $F_{c,2}(x_2).d \longleftarrow \max(h_2, j_2)$
$*Etape\ 4*$
si $h_1 < j_1$ **alors**
$F_{c,2}(x_2).p \longleftarrow 1$
sinon
$F_{c,2}(x_2).p \longleftarrow -1$
fin si
si $h_2 < j_2$ **alors**
$F_{c,1}(x_1).p \longleftarrow 1$
sinon
$F_{c,1}(x_1).p \longleftarrow -1$
fin si
fin si
fin pour
$*Fermer\ le\ polygone*$
si $p_{(N)1} \neq p_{(1)1}$ **et** $p_{(N)2} \neq p_{(1)2}$ **alors**
$c \longleftarrow c + 1$
$H = (h_1, h_2) \longleftarrow$ point d'intersection de la demi–droite $[P_{(1)}P_{(N)}$ avec le rectangle $ABCD$
$J = (j_1, j_2) \longleftarrow$ point d'intersection de la demi–droite $[P_{(N)}P_{(1)}$ avec le rectangle $ABCD$
$F_{c,1}(x_1).g \longleftarrow \min(h_1, j_1)$; $F_{c,2}(x_2).g \longleftarrow \min(h_2, j_2)$
$F_{c,1}(x_1).d \longleftarrow \max(h_1, j_1)$; $F_{c,2}(x_2).d \longleftarrow \max(h_2, j_2)$
si $h_1 < j_1$ **alors**
$F_{c,2}(x_2).p \longleftarrow 1$
sinon
$F_{c,2}(x_2).p \longleftarrow -1$
fin si
si $h_2 < j_2$ **alors**
$F_{c,1}(x_1).p \longleftarrow 1$
sinon
$F_{c,1}(x_1).p \longleftarrow -1$
fin si
fin si

celle construite dans le paragraphe précédent. La méthode modifiée, présentée dans cette section et résumée par l'algorithme 37, permet d'obtenir également une règle graduelle pour représenter un même polygone. Les contraintes sont définies plus localement que précédement puisque les sous–

Début

Etape 1. Vérifier que le segment traité n'est pas vertical ou horizontal.
Etape 2. Placer les points caractéristiques des fonctions d'appartenance en utilisant chaque paire de points consécutifs du polygone.
Etape 3. Choisir les pentes des fonctions d'appartenance en fonction du quadrant trigonométrique dans lequel se trouve le segment en considérant un système de référence centré sur le point d'origine du côté analysé dans le sens de parcours associé.

Fin

Procédure 3.2 – Algorithme amélioré – langage naturel

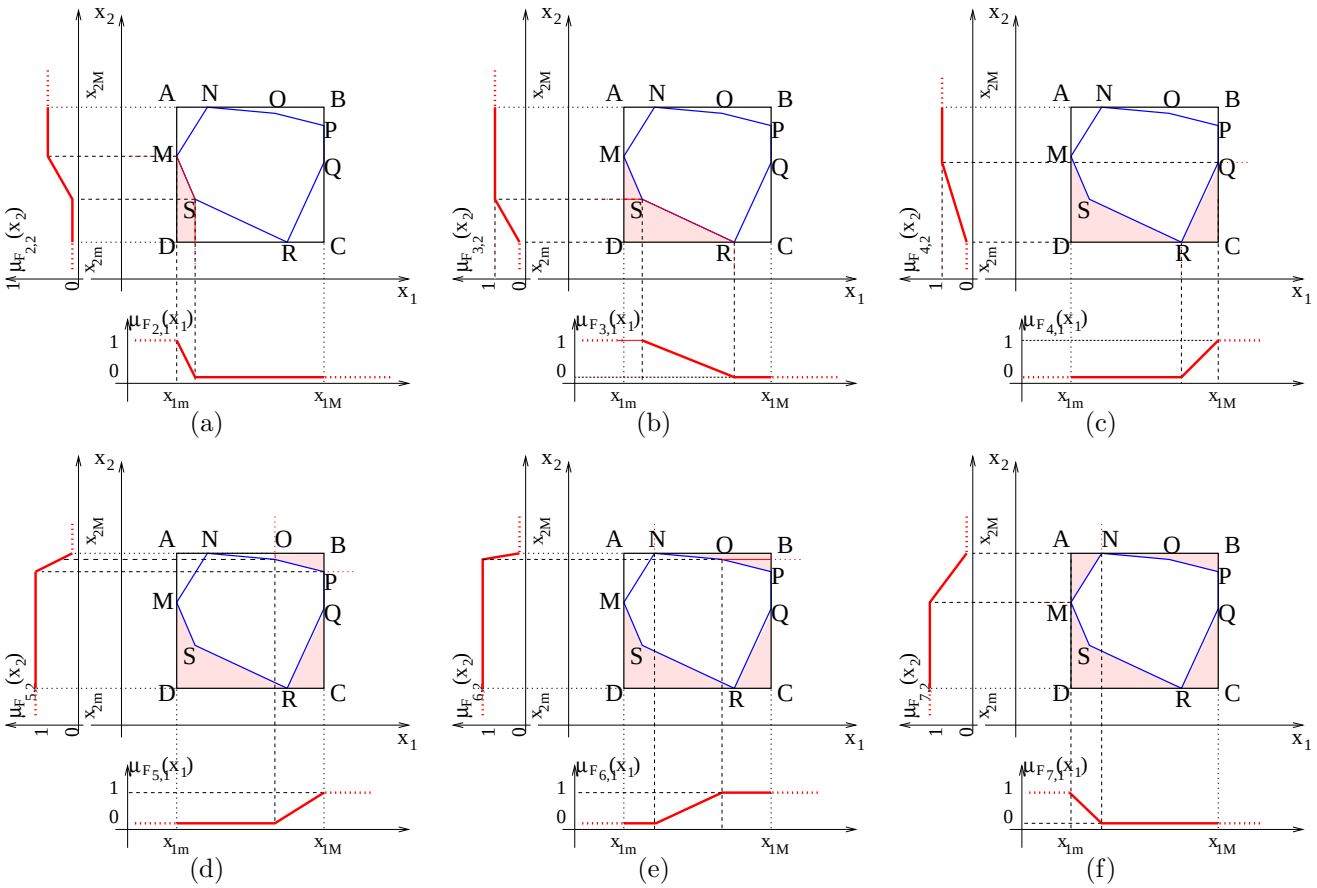


FIGURE 3.6 – Délimitation de l'espace réservé à la classe - approche améliorée

ensembles flous sont directement construits à partir des segments formant le polygone et non plus à partir de leur extension dans l'espace de travail. La figure 3.7 illustre cette démarche pour un segment $[AB]$ particulier. Ce principe de construction garantit que les zones interdites par l'algorithme 37 sont incluses dans celles interdites par l'algorithme 42. Autrement dit, l'algorithme 37 n'interdit pas de zones qui ne devrait pas l'être. Reste cependant à vérifier que cette nouvelle méthode est complète, c'est-à-dire qu'elle interdit bien toutes les zones qui doivent l'être.

La méthode proposée pour cette vérification est d'analyser les différentes successions possibles des côtés d'un polygone convexe. L'information pertinente pour l'approche proposée est liée à la position relative de deux côtés consécutifs dans un repère cartésien 2D centré sur leur point d'intersection (cf.

Algorithm 2 Algorithme de calcul des sous–ensembles flous pour l'approche améliorée

Entrée : le polygone P associé à la classe ; son rectangle englobant $ABCD$
Sortie : $\{F_{k,i}\}, \forall k = \overline{2, N_C + 1},\ \forall i \in \{1, 2\}$
Variables locales : i, k, c = l'ordre de la contrainte

$c \longleftarrow 1$
pour $k = 1$ **à** $N - 1$//Pour chaque point du polygone **faire**
$\overline{*Etape\ 1*}$
si $p_{(k)1} \neq p_{(k+1)1}$ **et** $p_{(k)2} \neq p_{(k+1)2}$ **alors**
$\overline{*Etape\ 2*}$
$c \longleftarrow c + 1$
$F_{c,1}(x_1).g \longleftarrow \min(p_{(k)1}, p_{(k+1)1})$; $F_{c,2}(x_2).g \longleftarrow \min(p_{(k)2}, p_{(k+1)2})$
$F_{c,1}(x_1).d \longleftarrow \max(p_{(k)1}, p_{(k+1)1})$; $F_{c,2}(x_2).d \longleftarrow \max(p_{(k)2}, p_{(k+1)2})$
$\overline{*Etape\ 3*}$
si $p_{(k)1} < p_{(k+1)1}$ **alors**
$F_{c,2}(x_2).p \longleftarrow 1$
sinon
$F_{c,2}(x_2).p \longleftarrow -1$
fin si
si $p_{(k)2} < p_{(k+1)2}$ **alors**
$F_{c,1}(x_1).p \longleftarrow 1$
sinon
$F_{c,1}(x_1).p \longleftarrow -1$
fin si
fin si
fin pour
$\overline{*Fermer\ le\ polygone*}$
si $p_{(N)1} \neq p_{(1)1}$ **et** $p_{(N)2} \neq p_{(1)2}$ **alors**
$c \longleftarrow c + 1$
$F_{c,1}(x_1).g \longleftarrow \min(p_{(N)1}, p_{(1)1})$; $F_{c,2}(x_2).g \longleftarrow \min(p_{(N)2}, p_{(1)2})$
$F_{c,1}(x_1).d \longleftarrow \max(p_{(N)1}, p_{(1)1})$; $F_{c,2}(x_2).d \longleftarrow \max(p_{(N)2}, p_{(1)2})$
si $p_{(N)1} < p_{(1)1}$ **alors**
$F_{c,2}(x_2).p \longleftarrow 1$
sinon
$F_{c,2}(x_2).p \longleftarrow -1$
fin si
si $p_{(N)2} < p_{(1)2}$ **alors**
$F_{c,1}(x_1).p \longleftarrow 1$
sinon
$F_{c,1}(x_1).p \longleftarrow -1$
fin si
fin si

figures 3.8 et 3.9).

Pour effectuer l'analyse, on fixe un côté $[AO]$ et on associe au système de référence le sens trigonométrique. Selon ce sens, il faut "interdire" la zone qui se situe à droite de ce segment. La droite support du segment $[AO]$ coupe le plan en deux demi–plans. Une première remarque est que, suivant le sens associé, un côté voisin au côté $[AO]$ ne peut se situer que dans le demi–plan gauche de la droite–support de $[AO]$, ce qui se traduit par un angle entre $[AO]$ et le segment suivant nécessairement supérieur à 180°. Ainsi, le segment $[OB]$ dans la figure 3.8 n'est pas un candidat

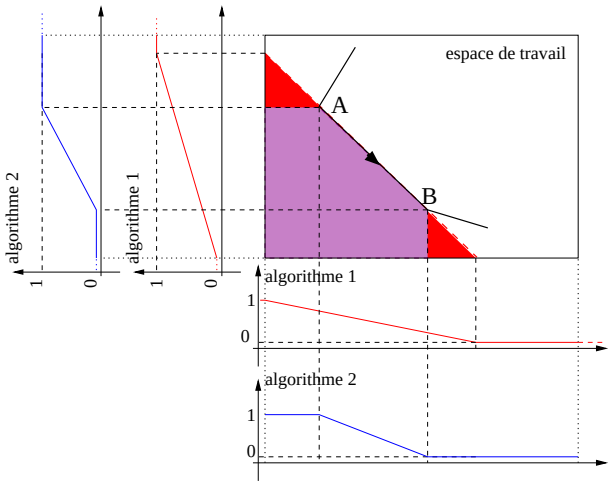


FIGURE 3.7 – Approche améliorée vs. approche de base

potentiel à la succession de $[AO]$. Les différents cas possibles de successions de côtés et les résultats de l'application des contraintes correspondantes sont présentés dans la suite, ainsi qu'une brève analyse de la signification de ces résultats.

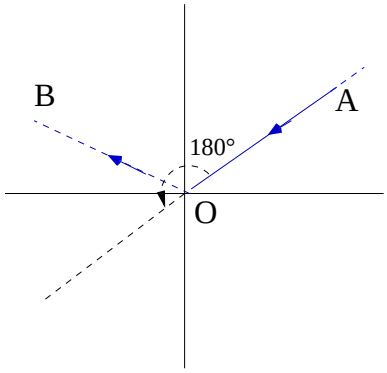
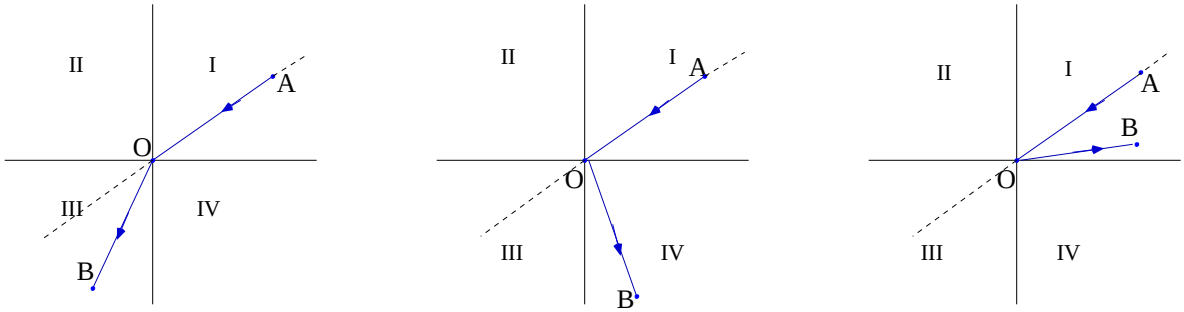


FIGURE 3.8 – Succession impossible de côtés

Les cas possibles sont présentés dans la figure 3.9 où les côtés voisins analysés sont les segments $[AO]$ et $[OB]$. On distingue en fait trois cas selon que le segment $[OB]$ est situé dans les quadrants III, IV ou I. Les quatre quadrants représentés sont dénommés comme dans la théorie trigonométrique de base : le quadrant I est le quadrant droit supérieur et les suivants sont numérotés dans le sens trigonométrique. Sur la figure 3.9(b) les deux côtés sont situés dans deux quadrants voisins. La figure 3.9(a) représente le cas où les deux côtés sont situés dans des quadrants opposés et sur la figure 3.9(c) on trouve le cas où les deux côtés successifs sont situés dans le même quadrant. Le comportement du système proposé selon ces positions est différent, comme expliqué dans les paragraphes suivants.

Comme le côté $[AO]$ a été choisi comme côté de référence et qu'il est commun aux cas présentés, la contrainte qui lui correspond est la même dans les trois cas. Cette contrainte est illustrée dans la figure 3.10.

La première image de la figure 3.11 montre la contrainte associée au côté $[OB]$ et son effet dans l'espace des attributs. La figure 3.11(b) représente la zone "interdite" associée au cas où les deux côtés sont situés dans des quadrants opposés. La remarque qui s'impose porte sur la redondance des zones "interdites". La totalité du quadrant II est interdite par chacune des deux contraintes. La



(a) Cas 1 (b) Cas 2 (c) Cas 3

FIGURE 3.9 – Successions possibles des côtés d'un polygone convexe

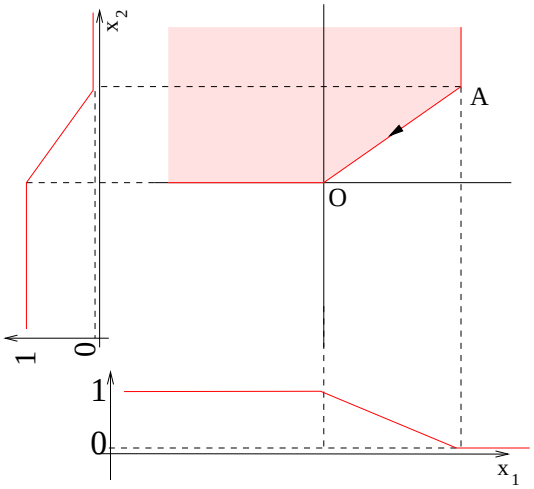
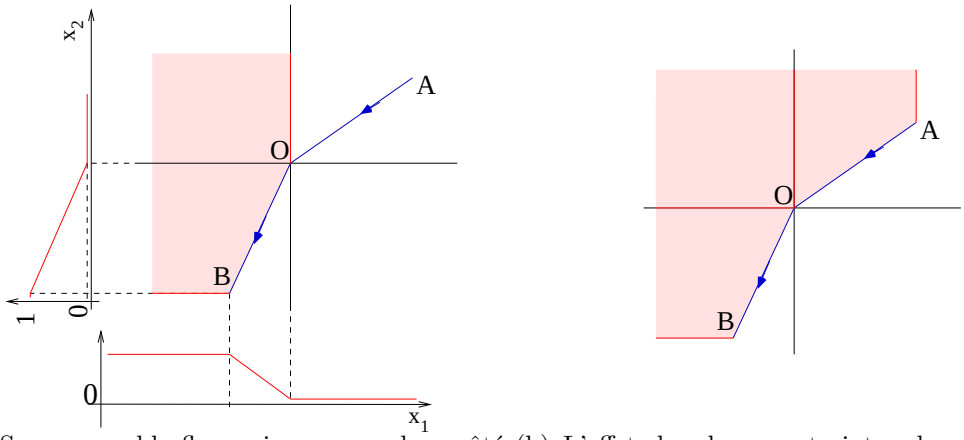


FIGURE 3.10 – Sous–ensemble flou qui correspond au côté $[AO]$

redondance peut être éliminée, mais le coût d'un tel traitement supplémentaire est trop élevé d'un point de vue du calcul et de l'analyse. De plus, les formes des fonctions d'appartenance ne seront pas les mêmes que les formes présentées, donc le formalisme devrait être modifié pour les inclure et interpréter. De toute manière, la redondance existe dans le système si on prend en considération la réduction de l'espace de travail au plus petit rectangle englobant.

La figure 3.12 représente la zone "interdite" associée au cas présenté dans la figure 3.9(b) après avoir aussi construit la contrainte associée au deuxième côté. La première image montre la contrainte associée au côté $[OB]$ et son effet dans l'espace des attributs. Comme précédemment, la zone "interdite" est marquée en couleurs. La figure 3.12(b) représente la zone interdite par les deux contraintes. La remarque qui s'impose est la jointure parfaite entre les deux zones "interdites" délimitées par les deux contraintes. Les deux zones ont une demi-droite commune, la demi-droite qui fait la démarquation entre les quadrants II et III.

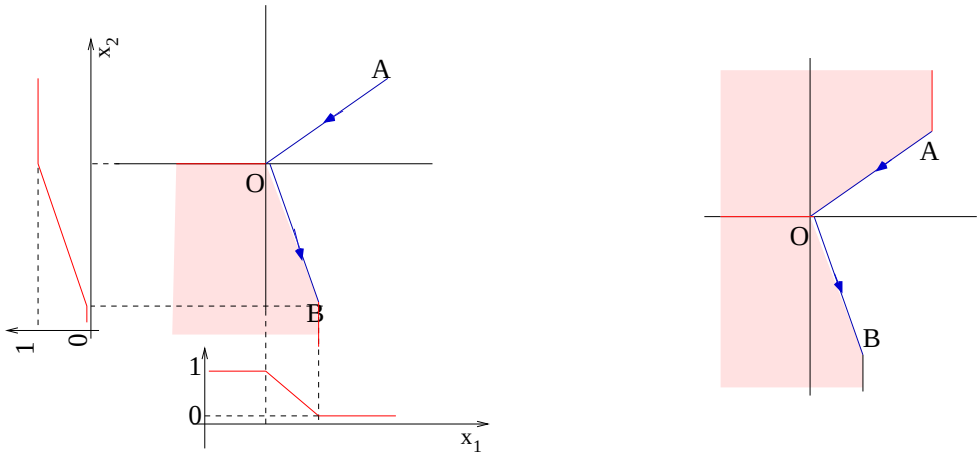
Par analogie aux deux cas précédents, la figure 3.13 représente les zones "interdites" associées aux deux côtés situés dans le même quadrant. La première image montre la contrainte associée au côté $[OB]$ et son effet dans l'espace des attributs et la deuxième la zone "interdite" associée aux deux contraintes. La remarque qui s'impose dans cette situation est l'existence d'une zone qui reste "permise" en dehors du polygone, en fait la totalité d'un quadrant (dans la situation présentée le quadrant III). Cette zone sera donc associée à la classe alors qu'elle devrait être "interdite" par les



(a) Sous–ensemble flou qui correspond au côté OB

(b) L'effet des deux contraintes dans l'espace des attributs

FIGURE 3.11 – Cas 1 – résultats



(a) Sous–ensemble flou qui correspond au côté OB

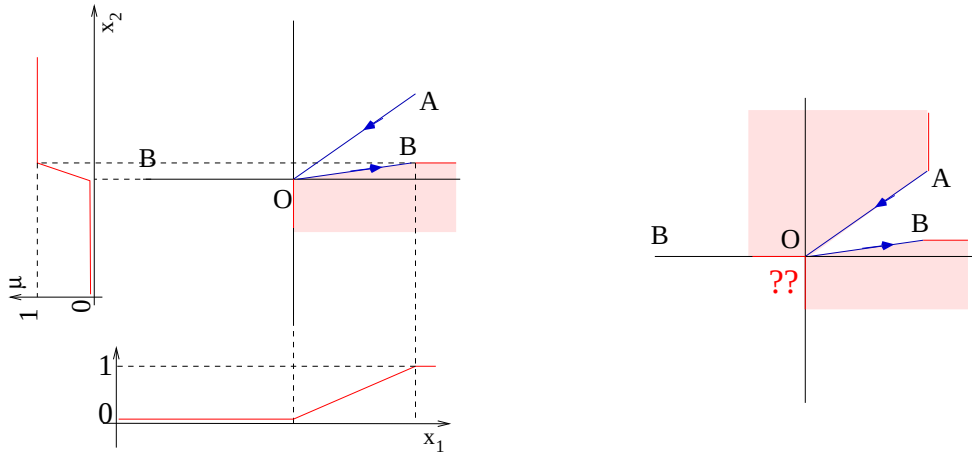
(b) L'effet des deux contraintes dans l'espace des attributs

FIGURE 3.12 – Cas 2 – résultats

contraintes associées aux deux côtés.

Dans le cas général, la méthode améliorée est donc incomplète et ne peut pas être utilisée directement pour "simplifier" les règles. Cependant, dans le cas particulier qui nous intéresse, deux contraintes spécifiques définissent l'espace de travail par le rectangle englobant le polygone. En considérant l'hypothèse de convexité du polygone, le point O, situé à l'intersection de deux côtés successifs placés dans le même quadrant, est forcément confondu à un sommet du rectangle englobant le polygone. Pour le cas illustré, le point O est identique au sommet situé en bas et à gauche du rectangle englobant. Dans cette situation, le quadrant III, qui constitue le "problème", sera éliminé par les contraintes qui définissent l'espace de travail.

Bien que le côté de référence $[AO]$ utilisé dans le raisonnement ait été supposé dans le quadrant I, la complétude de la méthode peut être démontrée de manière similaire pour un positionnement de $[AO]$ dans les trois autres quadrants. La conclusion est donc indépendante des quadrants où les côtés se situent et de leurs pentes. Une modification de ces paramètres correspond seulement à une



(a) Sous–ensemble flou qui correspond au côté OB

(b) L'effet des deux contraintes dans l'espace des attributs

FIGURE 3.13 – Cas 3 – résultats

rotation dans l'espace des attributs et respectivement à un réarrangement des sous–ensembles flous correspondants.

En conclusion, tout polygone convexe peut être délimité dans l'espace des attributs à l'aide d'une règle graduelle en utilisant le deuxième formalisme proposé. La condition qui doit être respectée est de se situer dans un espace de travail réduit au rectangle qui englobe ce polygone.

3.5 Interprétation

L'analyse du système effectuée dans les sections précédentes a été faite pour le contexte basique où on prend en considération l'existence de seulement deux attributs et une classe recherchée. Les contraintes exprimées dans la prémisse de règle permettent d'associer à chaque point de l'espace une valeur binaire dont la signification est l'appartenance (valeur 1) ou la non-appartenance (valeur 0) à la classe. Le but de cette section est d'étudier les différentes manières d'interpréter les contraintes d'abord d'un point de vue géométrique, puis ensembliste en utilisant les $\alpha - coupes$ et enfin analytiquement.

Une première "lecture" des contraintes porte sur leur signification géométrique. Pour faciliter l'interprétation des figures, une simplification des termes sera utilisée. Par "pente de la fonction d'appartenance" on entend la pente de la fonction d'appartenance dans sa partie "intéressante", de pente non-nulle. De manière similaire, par la "pente de la zone interdite", on indique la pente de la partie "intéressante" de la frontière de la zone "interdite", c'est–à–dire les parties non verticales et non horizontales.

La figure 3.14 présente un exemple d'influence de la pente de chacune des fonctions d'appartenance sur la forme de la zone "interdite" délimitée dans l'espace des attributs. Pour l'analyse présentée, les points "médians" des fonctions d'appartenance, c'est–à–dire correspondant à un degré d'appartenance de 0.5, ont été gardés et les pentes des fonctions d'appartenance ont été augmentées.

Dans le cas présenté dans la figure 3.14(a), la pente de la fonction d'appartenance appliquée au premier attribut est augmentée, alors que la fonction d'appartenance appliquée au deuxième attribut ne change pas. Le résultat est une augmentation de la pente de la zone "interdite", mais aussi

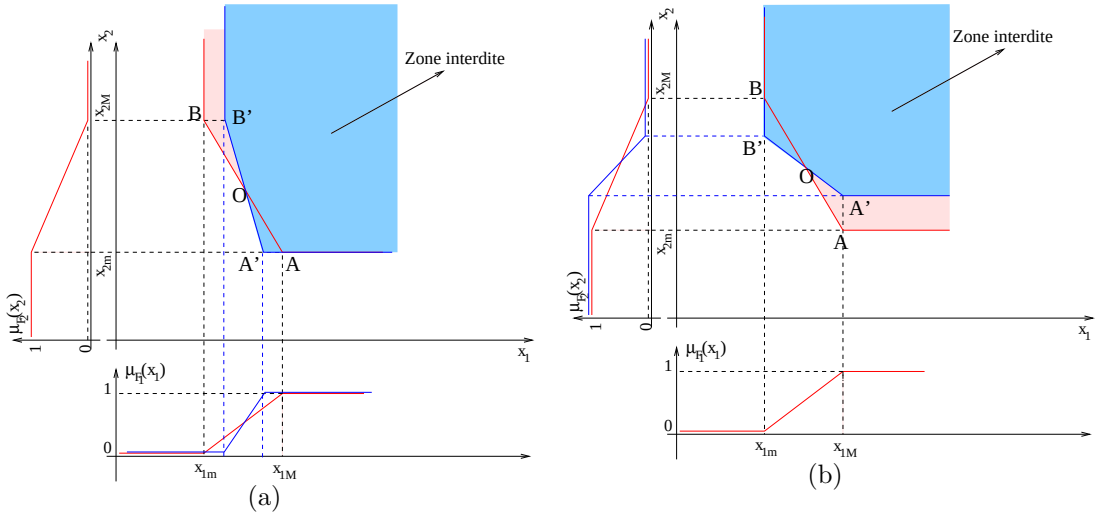


FIGURE 3.14 – L'influence du changement de la pente des fonctions d'appartenance

l'apparition d'une bande verticale "permise" sur la zone qui était "interdite" auparavant. Le point O, situé au milieu du segment $[AB]$ qui définit la zone "interdite" est commun aux deux situations. Il provient en fait des deux points "médians" des deux fonctions d'appartenance, qui n'ont pas bougé. L'effet d'une augmentation de la pente de la fonction d'appartenance appliquée au premier attribut n'est montré que pour une des quatre situations possibles présentées, mais les conclusions sont aussi valables pour les autre cas. La pente de la zone "interdite" augmente (se rapproche de la verticale) et la frontière verticale de cette zone est décalée en conséquence. L'ordonnée de la frontière horizontale de la zone "interdite" n'est pas affectée par le changement de la pente de la fonction d'appartenance sur le premier attribut.

La figure 3.14(b) présente l'effet de l'augmentation de la pente de la fonction d'appartenance appliquée au deuxième attribut. Contrairement au cas de la figure 3.14(a), la pente de la zone "interdite" diminue (se rapproche de l'horizontale). On remarque ici aussi l'apparition d'une bande "permise" dans la zone qui était "interdite" par l'utilisation d'une fonction d'appartenance de pente plus petite sur le deuxième attribut. Cette fois la bande est horizontale. Le point O du segment $[AB]$ est aussi commun aux deux situations. Comme dans le cas précédent, les conclusions restent valables pour les quatre autres cas possibles.

Le mélange entre les deux situations décrites revient à combiner les deux effets illustrés. Un cas particulier constitue la situation où les pentes des deux fonctions d'appartenance appliquées aux deux attributs augmentent simultanément dans les mêmes proportions. Dans ce cas–là, on retrouve une compensation des effets sur la pente de la droite qui donne la frontière de la zone interdite, donc le segment qui définit cette zone se trouve sur la même droite–support que le segment original, comme illustré dans la figure 3.15. D'un autre côté, les deux effets liés aux bandes horizontale et verticale supplémentaires permises restent. L'effet illustre en fait la différence entre les deux algorithmes de calcul des fonctions d'appartenance proposés pour une situation particulière, où le segment à représenter est situé à mi–distance entre les deux points d'intersection de sa droite–support avec les côtés du rectangle englobant.

Une autre interprétation possible est de considérer le degré d'appartenance du premier attribut comme une $\alpha - coupe$ appliquée sur la fonction d'appartenance du deuxième attribut. La figure 3.16 présente une telle interprétation. Si on considère le cas de $\alpha = 0.5$ présenté, il correspond en fait au point O dans les figures 3.14(a) et 3.14(b). L'interprétation peut alors se faire de manière

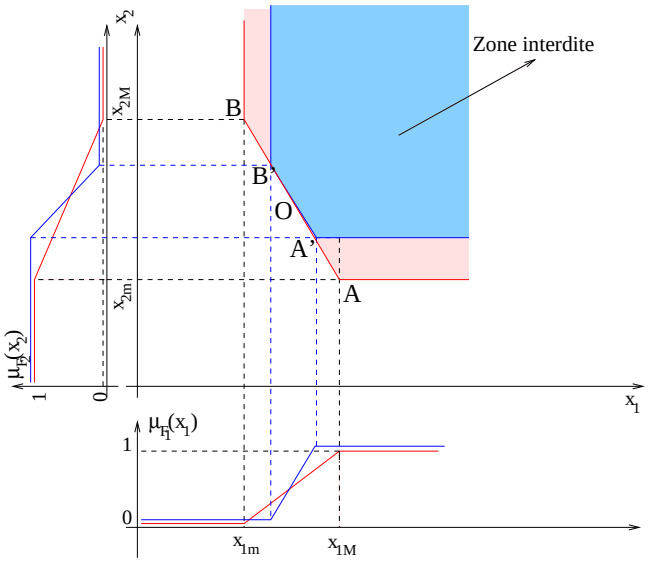


FIGURE 3.15 – L'influence des changements proportionnels des pentes des deux fonctions d'appartenance

immédiate : une $\alpha - coupe$ de 0.5 interdit tous les points qui ont un dégré d'appartenance à la fonction d'appartenance du deuxième attribut inférieur à 0.5. D'un point de vue géométrique dans l'espace des attributs, cela revient à interdire l'appartenance à la classe pour les points situés sur la demi–droite verticale qui passe par le point O. De façon similaire, chaque $\alpha - coupe$ correspond à une droite verticale dans l'espace des attributs. Comme les pentes des fonctions d'appartenance ne sont "intéressantes" que sur un intervalle borné, la signification n'est gardée que dans cet intervalle. D'un point de vue géométrique cet intervalle est justement celui défini par le segment de pente différente de 0° et 90°. En faisant varier α entre 0 et 1, on obtient tous les points–limites du segment $[AB]$. A $\alpha = 0$, correspond le point B si on considère l'égalité et la droite verticale qui passe par ce point et si on prend en considération la signification de l'implication. De même, en prenant $\alpha = 1$, on obtient la droite verticale qui passe par le point A.

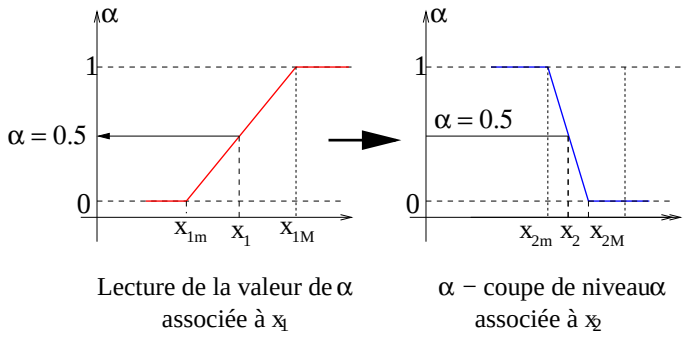


FIGURE 3.16 – L'interprétation par α-coupe

Une dernière interprétation analytique est également possible. Les zones "interdites" peuvent être caractérisées analytiquement en utilisant l'équation de la droite de pente non-nulle qui constitue sa frontière. En utilisant les quatre cas possibles décrits dans la figure 3.2, les équations de ces droites respectives peuvent être déduites. La droite–support du segment qui caractérise la zone "interdite", définie par les fonctions d'appartenance $F_{i,1}$ et $F_{i,2}$, est donnée par sa pente m_i et l'offset n_i, qui ont les valeurs suivantes :

$$m_i = F_{i,1}(x_1).p \times F_{i,2}(x_2).p \times \frac{x_{2M} - x_{2m}}{x_{1M} - x_{1m}} \tag{3.13}$$

$$n_i = \begin{cases} \frac{x_{2m}x_{1M} - x_{2M}x_{1m}}{x_{1M} - x_{1m}}, & si\ F_{i,1}(x_1).p \times F_{i,2}(x_2).p > 0 \\ \frac{x_{2M}x_{1M} - x_{2m}x_{1m}}{x_{1M} - x_{1m}}, & si\ F_{i,1}(x_1).p \times F_{i,2}(x_2).p < 0 \end{cases} \tag{3.14}$$

Le segment est limité sur cette droite par les deux valeurs limites sur les axes de représentation : x_{1m} et x_{1M} ou x_{2m} et x_{2M}. Les points situés dans les zones "interdites" satisfont alors les conditions données dans l'équation 3.15.

$$\begin{cases} \begin{cases} x_2 > x_{2m}, & x_1 < x_{1m} \\ x_2 > m \times x_1 + n, & x_1 \in [x_{1m}, x_{1M}] \end{cases} & Si\ F_{i,1}(x_1).p < 0\ et\ F_{i,2}(x_2).p < 0 \\ \begin{cases} x_2 < x_{2M}, & x_1 > x_{1M} \\ x_2 < m \times x_1 + n, & x_1 \in [x_{1m}, x_{1M}] \end{cases} & Si\ F_{i,1}(x_1).p > 0\ et\ F_{i,2}(x_2).p > 0 \\ \begin{cases} x_2 > x_{2m}, & x_1 > x_{1M} \\ x_2 > m \times x_1 + n, & x_1 \in [x_{1m}, x_{1M}] \end{cases} & Si\ F_{i,1}(x_1).p > 0\ et\ F_{i,2}(x_2).p < 0 \\ \begin{cases} x_2 < x_{2M}, & x_1 < x_{1m} \\ x_2 < m \times x_1 + n, & x_1 \in [x_{1m}, x_{1M}] \end{cases} & Si\ F_{i,1}(x_1).p < 0\ et\ F_{i,2}(x_2).p > 0 \end{cases} \tag{3.15}$$

3.6 "Commutativité" des attributs

Les implications qui sont à la base du système présenté ne sont pas commutatives. Inverser les deux attributs ne peut pas se faire de manière immédiate, par simple permutation des fonctions d'appartenance. Pourtant, ce changement des axes peut se faire, en tenant compte de quelques observations basiques.

En effet, inverser l'ordre des attributs revient à trouver la manière d'exprimer l'implication équivalente de $a \longrightarrow b$ sous une forme $f_b(b) \longrightarrow f_a(a)$. Autrement dit, il faut trouver le moyen de transférer le deuxième terme à la place du premier et inversement. A partir de la définition (3.3) on peut déduire l'équivalence décrite par l'équation (3.16).

$$a \longrightarrow b = \left\{ \begin{array}{ll} 1; & a \leq b \\ 0; & a > b \end{array} \right\} \iff a \longrightarrow b = \left\{ \begin{array}{ll} 1; & 1 - a \geq 1 - b \\ 0; & 1 - a < 1 - b \end{array} \right. \tag{3.16}$$

En utilisant la définition de l'implication de $Rescher - Gaines$ sur les degrés $1 - b$ et $1 - a$ on obtient l'équivalence parfaite de l'équation (3.17).

$$a \longrightarrow b \Leftrightarrow (1 - b) \longrightarrow (1 - a) \tag{3.17}$$

Si on revient au système de classification proposé, les valeurs a et b sont en fait les degrés d'appartenance des attributs $x_j, j \in \{1, 2\}$ selon les fonctions d'appartenance correspondantes. Si on note les variables $b' = 1 - b$ et $a' = 1 - a$, on obtient la relation de l'équation (3.18).

$$\begin{cases} a' = 1 - a = 1 - \mu_{F_{i,1}}(x_1) \\ b' = 1 - b = 1 - \mu_{F_{i,2}}(x_2) \end{cases} \tag{3.18}$$

Le changement de l'ordre des attributs revient alors à transférer les fonctions d'appartenance d'origine sur l'autre axe puis à en faire la négation conformément à l'équation (3.19), $F'_{i,j}(x_j)$ étant la fonction d'appartenance appliquée à l'attribut x_j dans le nouveau système de référence.

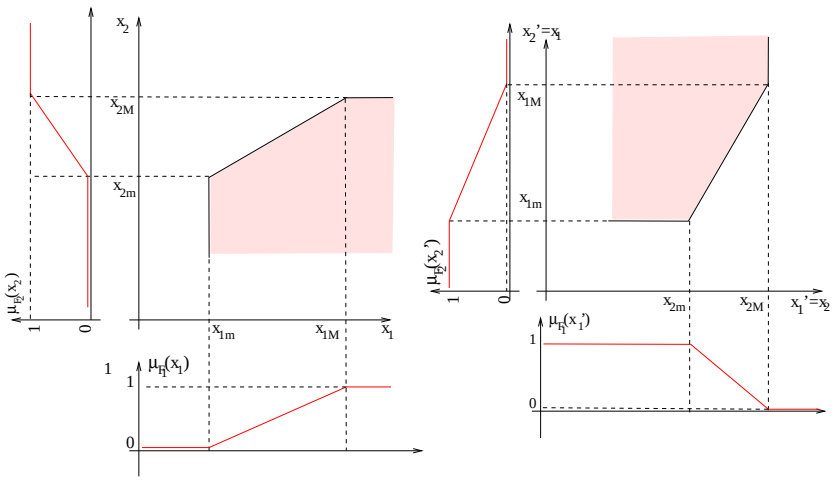
$$\mu_{F'_{i,j}}(x_j) = 1 - \mu_{F_{i,3-j}}(x_{3-j}) \tag{3.19}$$

Pour illustrer ces affirmations, on utilise le cas décrit par la figure 3.2(d). Le système de référence original est présenté dans la figure 3.17(a). L'équation de la zone "interdite" dans le système de référence modifié est donnée par (3.21), alors que la zone "interdite" du système de référence initial est donnée par l'équation (3.20). Dans l'équation (3.21) le changement de variable $\begin{cases} x'_1 = x_2 \\ x'_2 = x_1 \end{cases}$ a été utilisé.

$$\begin{cases} x_2 < x_{2M}, & x_1 > x_{1M} \\ x_2 < \frac{x_{2M}-x_{2m}}{x_{1M}-x_{1m}} \times x_1 + \frac{x_{2M}x_{1m}-x_{2m}x_{1M}}{x_{1m}-x_{1M}}, & x_1 \in [x_{1m}, x_{1M}] \end{cases} \tag{3.20}$$

$$\begin{cases} x'_2 > x_{1m}, & x'_1 < x_{2m} \\ x'_2 > \frac{x_{1M}-x_{1m}}{x_{2M}-x_{2m}} \times x'_2 + \frac{x_{1M}x_{2m}-x_{1m}x_{2M}}{x_{2m}-x_{2M}}, & x'_1 \in [x_{2m}, x_{2M}] \end{cases} \tag{3.21}$$

Comme montré dans la figure 3.17(b), la nouvelle zone est définie à l'aide des fonctions d'appartenance décroissantes, alors que la zone "interdite" d'origine est définie par des fonctions d'appartenance croissantes sur les deux attributs, comme illustré dans la figure 3.17(a).



(a) Système de référence initial (b) Système de référence inversé

FIGURE 3.17 – Illustration de la commutativité des attributs

En conclusion l'"inversion" des deux attributs peut se faire de manière immédiate en utilisant la formule (3.19). Les résultats des deux contraintes sont identiques. D'un point de vue logique, on retrouve en fait ici l'axiome de contraposition exploité dans les preuves par l'absurde.

3.7 Apprentissage des règles

3.7.1 Discussion générale sur les performances d'un système de classification basé sur des règles

Généralement, pour pouvoir obtenir une bonne classification sur un ensemble de données, les points qui constituent cet ensemble doivent respecter des contraintes de répartition. La qualité d'un système de classification dépend dans la plupart des cas de la compacité et de la séparabilité des classes. Si l'on considère le cas présenté dans la figure 3.18, il est évident que la réalisation d'une classification parfaite n'est pas possible même pour l'utilisateur humain. La cause provient de deux phénomènes :

- La grande variance de la classe 1 fait qu'il est impossible de la définir précisément. Une représentation précise ne peut se faire que si la classe est compacte et a une densité de points suffisamment élevée sur son domaine de définition, comme c'est par exemple le cas pour la classe 2.
- L'inclusion totale de la classe 2 dans la classe 1 ne permet pas de la distinguer. Un système "normal" de classification peut au mieux décider qu'"un point se trouve dans la classe 1, mais pas dans la classe 2" ou bien qu'"un point se trouve dans la zone de chevauchement des deux classes", mais la décision que "le point se situe bien dans la classe 2 et seulement dans cette classe" ne peut être qu'aléatoire. Une telle réponse ne peut pas être considérée comme correcte sans information a priori.

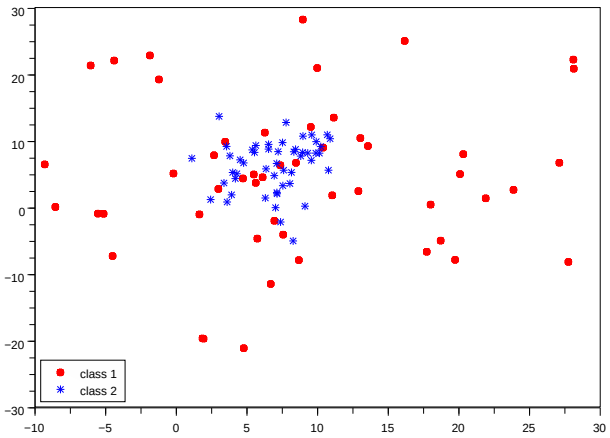


FIGURE 3.18 – Exemple de classes non séparables

Même dans cette situation, un système de classification peut délivrer une information pertinente sur un point qui doit être classé. Par exemple, si ce point est classé dans les deux classes possibles, la conclusion tirée par un utilisateur humain est que ce point doit être situé dans la zone commune des deux nuages de points. Cette information peut être utile pour certaines applications réelles. On peut par exemple associer un sens à une nouvelle classe composée des points placés dans la zone d'ambiguïté. Si les deux classes ont un sens physique, l'ambiguïté peut caractériser une région de passage entre les deux.

3.7.2 Le processus d'apprentissage

Pour illustrer le processus d'apprentissage, on a considéré dans un premier temps le problème de classification le plus simple : classifier un exemple par rapport à une classe. Dans cette situation la sortie d'un système de classification pour un point à classifier $X = (x_1, x_2)$ est une étiquette $l \in \{P, N\}$, avec la signification d'appartenance (P) ou non-appartenance (N) à la classe considérée. Une autre simplification utilisée pour la modélisation du cœur du système proposé est de considérer pour le moment le cas des ensembles de données bi-dimensionnelles, cas où le système traite des entités caractérisées uniquement par deux attributs.

Le cadre dans lequel les travaux se situent est l'apprentissage basé sur des exemples positifs (PEBL – Positive Example Based Learning, [73]), c'est–à–dire que l'on dispose des exemples qui appartiennent à la classe considérée et qui donc vont définir "l'intérieur" de la classe. Par contre, on ne dispose pas d'exemples qui n'appartiennent pas à la classe, donc on ne peut pas caractériser l'espace qui est en "dehors" de cette classe. Dans ce contexte, un système de classification sera une fonction $f : \mathcal{D} \rightarrow \{P, N\}$, où le domaine $\mathcal{D}$ est un domaine bi–dimensionnel. Dans un cadre numérique, le domaine $\mathcal{D}$ peut s'exprimer comme $\mathcal{D} = \Re^2$. Cette fonction doit représenter au mieux l'ensemble d'apprentissage disponible, donc elle doit associer la bonne étiquette à un maximum d'exemples possibles.

Dans le cadre simplifié proposé, le système réalise quelques étapes afin d'apprendre la règle associée à la classe :

1. à partir des points d'apprentissage, il construit la forme linéaire convexe qui définit la classe.
2. il définit les deux premières contraintes de la règle, qui réduisent l'espace de travail au plus petit rectangle qui englobe la forme définie au point 1.
3. il définit les contraintes associées aux côtés de la forme linéaire convexe.

A la fin des trois étapes, on dispose de la règle qui caractérise la classe dans l'espace 2D des attributs. Un point à classer recevra l'étiquette $l = P$ si d'un point de vue géométrique il est à l'intérieur de la forme associée à la classe et $l = N$ s'il est à l'extérieur de cette forme. Les deux dernières étapes ont été développées dans les sections précédentes. L'obtention de la forme convexe associée à la classe est détaillée dans la section suivante.

3.7.3 La construction du polygone qui définit la classe

Pour pouvoir construire la règle qui définit une forme utile pour le problème analysé, il faut d'abord déterminer la forme qui représente au mieux la classe. Le problème est donc de trouver pour une classe, dont on ne connaît que des points d'apprentissage lui appartenant, une forme géométrique linéaire convexe qui la représente. D'un point de vue formel et en tenant compte de l'approche, il est évident qu'une telle forme doit satisfaire quelques conditions de base :

– si l'ensemble d'apprentissage est parfait, tous ses points appartiennent effectivement à la classe et en sont représentatifs. Il faut donc que la forme définie inclue tous ces points.
– la surface qui est "permise" doit être suffisamment serrée autour des points afin d'inclure aussi peu de surfaces vides de points que possible.

La forme géométrique qui satisfait au mieux ces conditions est l'enveloppe convexe des points d'apprentissage. L'enveloppe convexe d'un ensemble de points est le plus petit ensemble convexe qui contient tous les points. Pour un domaine 2D borné (le cas étudié) l'enveloppe convexe est un polygone convexe. D'un point de vue intuitif, l'enveloppe convexe en 2D est une bande élastique

enroulée autour des points “extérieurs” du nuage. Quelques algorithmes ont été developpés afin de calculer cette enveloppe d'une manière optimisée. On peut des trouver sur les sites spécialisés :

– http ://www.softsurfer.com/Archive/algorithm_0109/algorithm_0109.html
– http ://www.cs.princeton.edu/ ah/alg_anim/version1/ConvexHull.html

L'algorithme choisi est l'algorithme “Graham Scan”.

Si l'on considère le cas présenté dans la figure 3.18 et que chaque classe est traitée indépendamment de l'autre, on associe à chacune des deux classes son enveloppe convexe, donc son polygone convexe. Le résultat est présenté dans la figure 3.19. Ainsi, l'analyse de l'ensemble d'apprentissage est limitée à l'extraction des points qui forment l'enveloppe convexe. Ces derniers sont ensuite la seule information utilisée pour obtenir les contraintes de la règle associée.

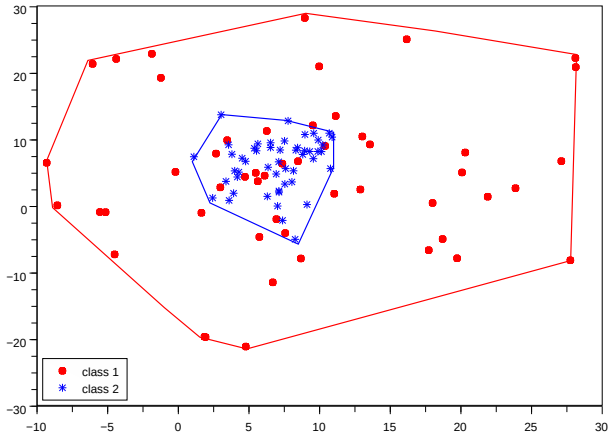


FIGURE 3.19 – Les enveloppes convexes des deux nuages de points

Dans tout ce chapitre, l'illustration de l'obtention des contraintes qui constituent une règle graduelle a justement été réalisée sur le polygone qui correspond à la deuxième classe.

Une solution au problème simple de classification présenté au début de ce chapitre a été proposé. Pourtant, les applications réelles ne consistent jamais à identifier seulement une classe à partir d'un ensemble d'apprentissage composé de points bidimensionnels. Le chapitre suivant présente la mise en œuvre réalisée afin de fusionner ces résultats de base dans un système complet de classification.

3.8 Résumé et conclusion

Ce chapitre a décrit le principe de base de l'approche proposée. Afin d'aider la lecture et d'augmenter la compréhension, un résumé paraît maintenant nécessaire.

En fait jusqu'à présent on a défini ce qu'on va appeller une “boîte élémentaire” du système développé, comme illustré dans la figure 3.20. Cette boîte constitue le composant de base d'un système de classification en phase d'exploitation. Elle correspond à une paire d'attributs et délivre la degré d'appartenance binaire (0 ou 1) à la classe C_i. Ce composant consiste en fait à appliquer la règle, de la forme donnée par l'équation (3.9) (page 57), obtenue pour cette classe et pour la paire d'attributs générique (x_k, x_j), aux valeurs (a_k, a_j) du point à classifier.

La sortie de la boîte est donnée par l'équation (3.10) (page 57). Comme l'implication de Rescher–

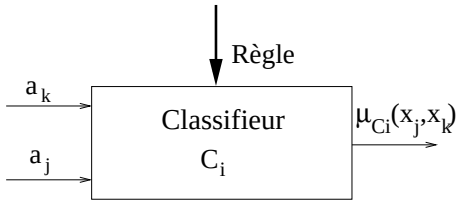


FIGURE 3.20 – Boîte élémentaire

Gaines est considérée, $\mu_{\Gamma_{(x_j,x_k)C_i}}(a_j, a_k)$ est une valeur binaire. Elle exprime un vote relatif à l'appartenance du point analysé à la classe considérée selon la paire d'attributs (x_k, x_j).

Le rôle de la boîte consiste à agréger conjonctivement les contraintes reliant les deux attributs pour la classe recherchée. La valeur unitaire de la sortie correspond à la situation où la paire des attributs vote pour l'appartenance et la valeur nulle correspond à la situation où elle vote pour la non–appartenance du point à la classe C_i. D'un point de vue géométrique la valeur unitaire correspond à la situation où le point donné par les deux attributs analysés se situe à l'intérieur de la forme géométrique décrite par les contraintes de la règle correspondante, notamment à l'intérieur de l'enveloppe convexe du nuage de points d'apprentissage de la classe C_i dans l'espace des attributs (x_k, x_j).

L'utilisation de cette "boîte élémentaire" de classification dans un système complexe est décrite dans le chapitre suivant.

Chapitre 4

Système de classification développé

Afin de tester le principe de base décrit dans le chapitre précédent, des applications réelles de classification ont été considérées. Deux problèmes principaux ont dû être abordés dans ce contexte :

- Les données d'apprentissage ne sont pas toujours assez "propres" et introduisent de l'information fausse ou non–pertinente dans le système d'apprentissage. Deux types de traitement sont proposés afin de réduire l'effet des données "sales".
- Le principe présenté s'applique au cas trivial des données caractérisées par deux attributs dans un contexte de classification binaire (il faut décider si l'exemple à classer appartient ou non à une classe). Il est donc nécessaire d'agréger ces résultats afin d'obtenir un système capable de traiter des données multi–dimensionnelles et de distinguer entre plusieurs classes.

4.1 Traitements de l'ensemble d'apprentissage

Dans cette section, on considére à nouveau la situation de base où l'on traite des ensembles d'apprentissage caractérisés par deux attributs. Les analyses proposées seront ensuite étendues dans la section suivante. Comme on se situe dans le contexte particulier de la classification supervisée, le problème de la qualité et de la taille de l'ensemble d'apprentissage est essentiel pour obtenir des résultats pertinents. Cette dépendance de la qualité de la classification à l'ensemble d'apprentissage peut être évoquée sous plusieurs aspects :

- L'exactitude de l'ensemble d'apprentissage : un point d'apprentissage "faux", qui se situe d'un point de vue géométrique loin de la plupart des autres points peut avoir des effets négatifs importants sur les taux de classification. D'un point de vue de l'approche proposée, l'existence de ces points "faux" revient à associer la classe à un polygone beaucoup plus large que le polygone réellement nécessaire. La forme du polygone est ainsi très affectée par l'existence de ces points. Le résultat de la chaîne de classification est l'allocation de beaucoup de points de l'espace des attributs à la classe, alors qu'ils sont situés loin du "cœur" du nuage des points d'apprentissage. Si l'on considère aussi l'existence des autres classes dans l'espace des attributs, l'ambiguïté peut être artificiellement augmentée à cause de ces points.
- La distribution interne des points d'apprentissage : la méthode donne de bons résultats tant que les points d'apprentissage d'une même classe décrivent une forme compacte dans l'espace 2D considéré. Deux problèmes peuvent pourtant apparaître selon la forme des nuages de points analysés. Le premier est lié à l'existence de plusieurs composantes connexes dans le nuage de points associé à une même classe. Comme le système traite des attributs qui ont des valeurs

continues, il sera nécessaire de définir proprement la notion de connexité. L'effet d'une situation où le nuage de points est non–connexe est l'inclusion des zones vides entre les diverses composantes dans l'enveloppe de la classe, ce qui diminue la fiabilité de la classification. Un autre problème lié à la distribution des points est l'organisation des points dans des formes atypiques. Si la forme du nuage de points est concave (comme dans le cas des nuages en forme de **U** ou **L**) le système inclut la zone vide de points de la concavité dans la forme associée à la classe. Dans cette situation, il faut également définir la notion de concavité d'un ensemble de points.

Ces problèmes apparaissent souvent dans le cas des applications réelles. Les points d'apprentissage pour ces applications proviennent de systèmes de mesure qui sont susceptibles d'introduire des erreurs, et la présence de points d'apprentissage "faux" est presque inévitable. De plus, dans le cas des applications réelles, une classe donnée peut être composée par plusieurs sous–classes, chacune avec ses propres caractéristiques. Dans l'espace des attributs, cela se traduit par la distribution des points d'apprentissage de cette classe en plusieurs composantes connexes. Dans le cas idéal, ces composantes devraient être traitées séparément et non pas être allouées au même polygone englobant.

La suite de cette section présente deux traitements proposés afin de résoudre ou réduire ces problèmes. Il faut préalablement souligner que les algorithmes proposés n'ont pas comme but de régler de manière exhaustive ces inconvénients, mais de les réduire, et donc d'augmenter au maximum la qualité et la fiabilité de la classification avec un coût minimum en puissance et temps de calcul.

4.1.1 Epuration de l'ensemble d'apprentissage

Les points qui sont pris en considération dans le calcul des polygones associés à chaque classe sont les points d'apprentissage "extrêmes". De ce fait, les points d'apprentissage non–pertinents ont une influence importante sur la qualité de la classification quand ils sont situés en périphérie du nuage de points ou, pire encore, quand ils sont situés loin de celui-ci. La caractéristique principale de ces points, qui permet leur identification, est leur fréquence réduite par rapport aux autres points qui sont représentatifs pour la classe recherchée.

Par exemple, pour les problèmes de classification à partir d'images, qui ont fait l'objet de la plupart des applications développées dans cette thèse, les ensembles d'apprentissage proviennent de zones de référence pointées par des experts comme étant représentatives des classes recherchées. Si ces régions sont affectées par un bruit d'une intensité peu élevée, le nombre de points affectés par ce bruit (les points d'apprentissage "faux") reste assez faible, donc leur fréquence dans l'ensemble d'apprentissage est également faible.

La figure 4.1 représente un ensemble d'apprentissage généré aléatoirement afin d'illustrer le problème. On considère que les points représentés définissent une même classe. Un faible pourcentage de ces points sont intentionnellement situés très loin du nuage principal. L'effet de ces points est très visible dans la figure 4.1, où le polygone résultant est aussi représenté. A cause d'un unique point d'apprentissage aberrant, ce polygone est beaucoup trop large et inclut une large surface vide de points.

La méthode proposée par le système afin d'éliminer ces points est basée justement sur cette hypothèse de faible fréquence des points d'apprentissage. Elle utilise comme information de base la fréquence de chaque point–candidat et la fréquence des points d'apprentissage qui se trouvent dans son voisinage. Les points d'apprentissage qui sont caractérisés par une fréquence d'apparition faible et qui sont situés dans une région de l'espace des attributs qui contient peu de points d'apprentissage sont considérés atypiques pour la classe recherchée. Ils seront donc éliminés de l'ensemble

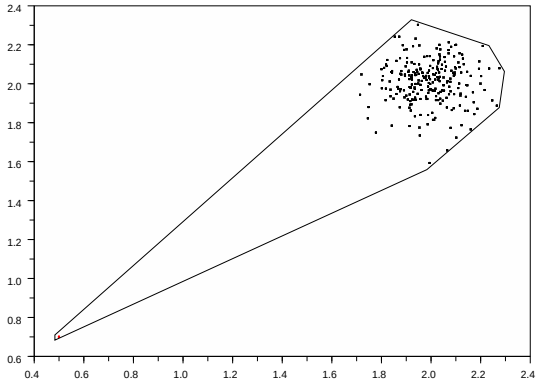


FIGURE 4.1 – L'influence des points d'apprentissage "faux" sur la représentation d'une classe

d'apprentissage.

La mise en œuvre d'une telle stratégie nécessite que soit préalablement définie la notion de voisinage dans l'espace des attributs considérés. L'espace est donc découpé en cellules selon un maillage rectangulaire. Tous les points se trouvant dans une même cellule seront considérés comme voisins entre eux. L'ensemble de cellules forment un réseau discret dont la résolution (taille des cellules) doit être paramétrée, comme montré sur la figure 4.2. A chaque cellule du réseau discret est associé le nombre de points d'apprentissage contenus dans la cellule.

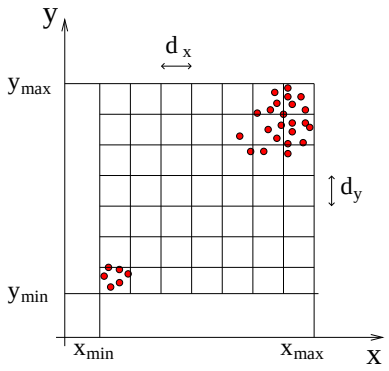


FIGURE 4.2 – Discrétisation de l'espace des attributs

Un histogramme h est ainsi calculé sur le réseau discret construit. Par la suite, les cellules ne contenant aucun point seront appellées des cellules "vides". Toutes les autres, qui contiennent au moins un point, seront identifiées par le terme de cellule "pleine". La densité moyenne $\overline{d}$ de remplissage des cellules pleines est évaluée à l'aide de l'équation (4.1).

$$\overline{d} = \frac{Nb\ points}{Nb\ cases\ pleines} \tag{4.1}$$

Le filtrage consiste ensuite à éliminer les points d'apprentissage qui se situent dans les cellules où le nombre de points est faible par rapport à la densité moyenne, c'est–à–dire en dessous d'un seuil exprimé en fonction de $\overline{d}$. Les cases dont la densité de points est faible, définies par l'équation (4.2), sont vidées de leurs points et deviennent des cases vides.

$$h(k,l) < \frac{\overline{d}}{\alpha},\ \forall k,l \in \text{au réseau discret} \tag{4.2}$$

Ce filtrage, rendu adaptatif par l'utilisation de $\overline{d}$, permet d'avoir une épuration maîtrisée de l'ensemble de données. Une valeur de α fixée à 2 a montré de bons résultats à la fois sur des données synthétiques et réelles.

Cette approche nécessite également le paramétrage de la construction du réseau discret, c'est–à–dire le pas de discrétisation de l'espace. Il est possible d'utiliser un pas fixe appliqué dans tous les cas, mais cette approche simpliste trouve rapidement ses limites. En effet, il est très difficile de trouver une résolution unique qui satisfasse toutes les applications. La pertinence de la discrétisation va dépendre de l'étendue des points dans l'espace des attributs ainsi que de leur distribution locale. Il a donc été décidé de paramétrer le choix de la résolution et de la rendre adaptative. La solution simple qui a été choisie consiste à s'appuyer sur la plus petite différence de coordonnées sur les deux composantes (notées d_{minx} et d_{miny}) entre deux points distincts (non–superposés) de l'ensemble d'apprentissage. Les dimensions des cellules seront alors un multiple de ces écarts minimums : $d_x = \beta d_{minx}, d_y = \beta d_{miny}$.

L'algorithme de principe du filtrage se résume finalement par les étapes présentées dans la procédure 4.1.

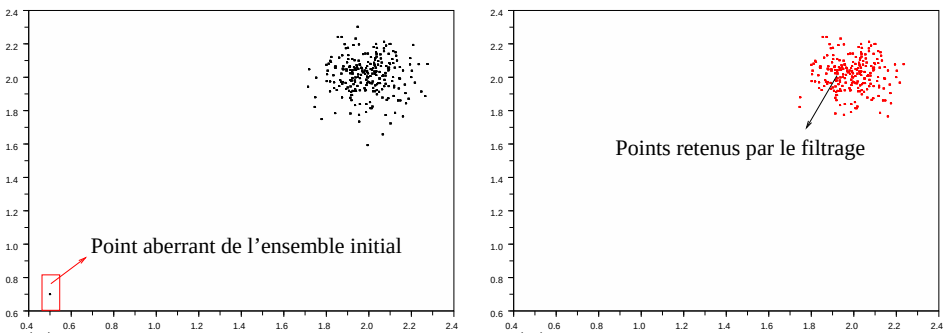
Début
- Etape 1. Calcul des paramètres du réseau discret (d_x, d_y).
- Etape 2. Comptage des points dans chaque cellule et calcul de la densité moyenne $\overline{d}$.
- Etape 3. Elimination des points appartenant à des cellules qui ont une densité de remplissage trop faible.

Fin

Procédure 4.1 – Algorithme de nettoyage des points aberrants

La méthode complète est quant à elle détaillée par l'algorithme 27. L'épuration de l'ensemble d'apprentissage, réalisée à la fois de façon adaptative et empirique, apporte un compromis satisfaisant entre robustesse et temps de calcul.

La figure 4.3 illustre graphiquement l'épuration obtenue sur l'ensemble d'apprentissage initialement présenté à la figure 4.1. Sur cet exemple, 17 points ont été supprimés sur les 194 points initiaux (soit 8.76% des points).



(a) Ensemble d'apprentissage initial (b) Ensemble d'apprentissage filtré

FIGURE 4.3 – Application de l'algorithme d'épuration des points aberrants

Les expérimentations réalisées sur l'ensemble d'apprentissage de la figure 4.3(a) pour différentes dimensions du réseau discret, c'est–à–dire différentes valeurs de β, montrent (Tab. 4.2) un point d'inflexion du nombre de points supprimés. Par la suite, un pas d'échantillonnage trois fois supérieur aux distances d_{minx} et d_{miny} est utilisé, c'est–à–dire $\beta = 3$.

Algorithm 3 Epuration de l'ensemble d'apprentissage

Entrée : l'ensemble d'apprentissage $(a_{ix}, a_{iy}), i = \overline{1, Nb_{pts}}$
Sortie : l'ensemble d'apprentissage nettoyé $(a_{jx}, a_{jy}), j = \overline{1, Nb_{reduit}}$
Variables locales :

- k,l
- d_{minx}, d_{miny} : la plus petite différence de coordonnées > 0 entre les points de l'ensemble d'apprentissage
- a_{xmin}, a_{ymin} : la plus petite valeur de l'ensemble sur l'axe x/y
- a_{xmax}, a_{ymax} : la plus grande valeur de l'ensemble sur l'axe x/y
- $step_x, step_y$: le pas de discrétisation sur l'axe x/y
- N_x, N_y : le nombre des cellules obtenues sur l'axe x/y
- density[k,l] : la densité de points de la cellule (k,l)
- Nb_{plein} : le nombre de cases pleines
- $\overline{d}$: la densité moyenne des points d'apprentissage dans les cases pleines

$\overline{*Etape\ 1*}$
Rechercher d_{minx} et d_{miny}
Rechercher a_{xmin}, a_{ymin}, a_{xmax} et a_{ymax}
$step_x \longleftarrow 3 \times d_{minx}$; $step_y \longleftarrow 3 \times d_{minx}$ /*$\beta = 3$*/
$N_x = |\frac{a_{xmax} - a_{xmin}}{step\_x}|$; $N_y = |\frac{a_{ymax} - a_{ymin}}{step\_y}|$

$\overline{*Etape\ 2*}$
/*Calcul de la densité des points dans chaque cellule*/
pour $k = 1$ **à** N_x **faire**
  pour $l = 1$ **to** N_y **faire**

$$density[k,l] \longleftarrow Card\{(a_{ix}, a_{iy}) / a_{ix} \in [a_{xmin} + (k-1) \cdot step_x, a_{xmin} + k \cdot step_x), \\ a_{iy} \in [a_{ymin} + (l-1) \cdot step_y, a_{ymin} + l \cdot step_y)\}$$

  fin pour
fin pour
Comptabiliser Nb_{plein}
$\overline{d} \longleftarrow \frac{Nb_{pts}}{Nb_{plein}}$
$\overline{*Etape\ 3*}$
$\overline{Nb_{reduit} \longleftarrow 0}$
/*Recherche des cellules à conserver et copie des points*/
pour $k = 1$ **à** N_x **faire**
  pour $l = 1$ **à** N_y **faire**
    si $density[k,l] \geq \frac{\overline{d}}{2}$ **alors**
      pour $j = Nb_{reduit}$ **à** $Nb_{reduit} + density[k,l]$ **faire**
        copier en (a_{jx}, a_{jy}) les points de la cellule [k,l]
      fin pour
      $Nb_{reduit} \longleftarrow Nb_{reduit} + density[k,l]$
    fin si
  fin pour
fin pour

4.1.2 Identification des composantes connexes

La compacité du nuage de points associé à une classe peut aussi influencer la qualité de la classification. Si les points sont répartis dans plusieurs régions compactes, un système optimal devrait

TABLE 4.2 – Impact du réseau discret sur l'épuration

Pas d'échantillonnage	d_{min}	$2 \times d_{min}$	$\mathbf{3 \times d_{min}}$	$4 \times d_{min}$	$5 \times d_{min}$
Points supprimés(%)	0	7.96	**8.76**	8.76	7.17

traiter ces composantes indépendamment les unes des autres. La figure 4.4 présente un tel cas, où les points d'apprentissage relatifs à une unique classe sont concentrés dans deux nuages connexes. Dans cette situation le polygone englobant inclut d'importantes zones vides de points [30].

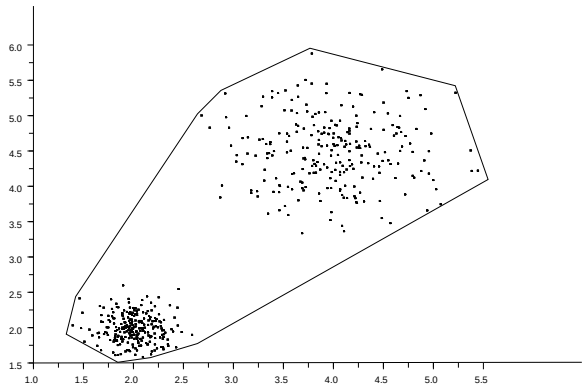


FIGURE 4.4 – L'influence d'une répartition non–connexe des points d'apprentissage

Il faut souligner que ce type de problème, même s'il reste spécifique, peut se retrouver dans des applications réelles. La cause de l'apparition de ce phénomène est le fait que parfois une classe recherchée peut être composée de plusieurs sous–classes qui ont des comportements différents et qui sont donc caractérisées par des nuages bi–localisés.

Les contraintes liées à ce pré–traitement restent les mêmes que précédemment, à savoir limiter la complexité pour garder des temps de calcul raisonnables, l'objectif étant avant tout de réduire l'effet de la non–connexité et non de trouver le nombre optimal de composantes connexes.

La détection des composantes connexes peut être vue comme un problème de classification non–supervisée qui a pour but d'obtenir une partition "naturelle" des points d'apprentissage. Le nombre de composantes connexes (sous–classes) n'étant pas connu a priori, la première approche choisie a été l'algorithme de "competitive agglomeration" [50] dans l'objectif d'obtenir une partition correcte des données avec un nombre restreint de sous–classes. Malheureusement, les résultats obtenus dans [30] ont mis en évidence que la partition floue générée n'était pas directement exploitable pour déterminer les différentes composantes connexes et qu'il était encore nécessaire de réduire le nombre de sous–classes obtenu. Dans ce contexte, vu la complexité calculatoire de la méthode et le nombre important de paramètres à régler, une approche plus pragmatique a été adoptée. Celle–ci se décompose en deux étapes, la première réalisant un partitionnement net initial des données, la seconde raffinant ce partitionnement par des traitement ad–hoc. La partition initiale est alors générée par une méthode empirique simple, à savoir le "basic isodata" [44, 63]. Cet algorithme est une variante de l'algorithme des plus proches voisins décrit dans le chapitre II de cette thèse. Son principe est donné dans la procédure 4.3.

Le nombre de sous–classes est fixé au nombre de points d'apprentissage divisé par 10 avec une borne minimale de 2 et maximale de 10. Dans l'hypothèse d'une distribution uniforme des exemples, ce choix impose au moins 10 points d'apprentissage par sous–classe. Un désavantage général de

Début
Etape 1. Le nombre a priori des classes est fixé à $C = min(10, max(2, Nbpts/10))$.
Etape 2. Les points sont répartis aléatoirement dans les C classes.
Etape 3. Les "centres" des classes ainsi définies sont calculés.
Etape 4. Pour chaque point, on détermine le centre de classe le plus proche et on le réaffecte à la classe correspondante.
Etape 5. Si au moins un point a changé de classe, on recalcule les centres.
Etape 6. Les étapes 4. et 5. sont répétées jusqu'à ce qu'un nombre maximal d'itérations soit atteint ou qu'aucun point n'ait changé de classe.
Fin

Procédure 4.3 – Principe de l'algorithme "basic isodata"

cet algorithme est qu'il a tendance à devenir lourd en terme de temps d'exécution et ressources calculatoires. Comme dans le contexte de travail proposé, on n'a pas besoin que le résultat soit d'une qualité exceptionnelle, le critère d'arrêt a été fixé de façon très stricte. Ainsi, le processus est arrêté après quelques itérations, même si les sous–classes identifiées ne sont pas stabilisées. Par conséquent, l'algorithme est exécuté rapidement et sans une consommation importante de ressources de calcul. Même si la répartition obtenue par cet algorithme n'est pas optimale, elle est raffinée dans une deuxième étape.

Dans cette deuxième étape, les points initialement répartis dans les différents nuages sont ré–étiquetés si besoin est. Pour cela, on ordonne tout d'abord les nuages selon leur cardinal et on calcule pour chaque nuage n identifié sa distance interne $d_{intranuage}(n)$. Cette distance est définie comme la moyenne des distances minimales entre les points qui forment le nuage. Son expression est donnée par l'équation (4.3)

$$d_{intranuage}(n) = \frac{\sum_{i=1}^{N} \min_{j,j\neq i} \sqrt{(a_{ix} - a_{jx})^2 + (a_{iy} - a_{jy})^2}}{N}, \text{ avec N=Cardinal(n)} \tag{4.3}$$

Ensuite, les points qui composent chacun des nuages sont à nouveau analysés de façon à procéder à leur éventuel réétiquetage. Pour tout point i du nuage k, on détermine son plus proche voisin parmi les points des nuages de cardinal supérieur. Soit d_{min} la distance à ce plus proche voisin et l le nuage auquel il appartient. Si d_{min} est inférieure à la distance interne des deux nuages impliqués ($d_{intranuage}(k)$ et $d_{intranuage}(l)$), le point i est réétiqueté comme appartenant au nuage l. Cette stratégie de réallocation est répétée jusqu'à ce qu'il n'y ait plus de point qui change de nuage ou qu'un nombre maximal d'itérations soit atteint. Le résultat est la réduction du nombre de nuages connexes tout en respectant leur distance interne $d_{intranuage}$.

Le principe de l'algorithme est décrit en langage naturel dans la procédure 4.4, puis détaillé dans l'algorithme 31.

La procédure a été appliquée sur l'ensemble d'apprentissage présenté dans la figure 4.4. Les résultats sont montrés dans la figure 4.5. Les deux nuages visuellement isolés sont identifiés correctement (les points de chaque nuage identifié sont représentés avec une couleur).

La procédure proposée n'est pas une méthode de classification non–supervisée, elle a comme seul but d'identifier les composantes connexes d'un ensemble d'apprentissage qui sont clairement séparées, afin d'aider le système de classification à base de règles graduelles proposé. Elle est basée sur une

Début

Etape 1. Initialisation des nuages par l'algorithme "basic isodata".
Etape 2. Tri des nuages par ordre croissant de cardinal.
Etape 3. Calcul des distances intranuages.
Etape 4. Réétiquetage des points :
    Pour chaque nuage k :
        Pour chaque point i du nuage k :
            point j ⟵point le plus proche du point i parmi
                les points des nuages $> k$
            d_{min} ⟵distance entre les points i et j
            nuage l ⟵nuage auquel appartient le point j
            Si $d_{min} < d_{intranuage}(k)$ et $d_{min} < d_{intranuage}(l)$ le point
                i est réétiqueté dans le nuage l
Etape 5. Répéter les étapes 2, 3 et 4 jusqu'à ce que plus aucun point ne change
        de nuage ou que le nombre maximal d'itérations soit atteint

Fin

Procédure 4.4 – Identification des composantes connexes – principe

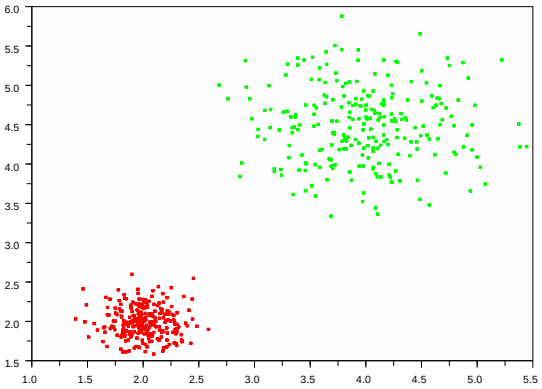


FIGURE 4.5 – Identification des nuages connexes

stratégie ad–hoc, sans doute améliorable d'un point de vue complexité et convergence.

4.2 Architecture d'un système de classification à base de composants élémentaires

Les problèmes réels de classification n'ont pas la simplicité du cas présenté. Les exemples à classifier sont décrits par plusieurs attributs et il faut distinguer entre plusieurs classes. Le chapitre précédent a montré l'apprentissage d'une règle qui assure la classification d'un point dans un espace de deux attributs pour une seule classe. Plusieurs problèmes doivent être traités afin de pouvoir appliquer le système proposé sur les cas réels de classification :

- Etendre le principe utilisé au cas des points caractérisés par plusieurs attributs pour établir l'appartenance ou la non-appartenance à une seule classe.
- Prendre en considération l'existence de l'ensemble des classes et gérer leurs possibles superpositions.

Algorithm 4 Identification des composantes connexes associées à une même classe

Entrée : l'ensemble d'apprentissage $a_i = (a_{ix}, a_{iy}), i = \overline{1, Nb_{pts}}$
Sortie : l'ensemble d'apprentissage séparé en composantes connexes :
$cc_k = \{a_i / label(a_i) = k\}$; $label(a_i)$ = l'étiquette du point $a_i = (a_{ix}, a_{iy})$
Variables locales :
- l : compteur
- $d_{intranuage}(k), k = \overline{1, C}$: la distance moyenne du nuage k
- $dmin$: la distance minimale d'un point à un autre point situé dans un nuage voisin

$\overline{*Etape\ 1*}$
C $\longleftarrow min(10, max(2, Nb_{pts}/10))$
Algorithme Basic ISODATA [44] $\Longrightarrow cc_k,\ k = \overline{1, C}$
modif $\longleftarrow$ VRAI
$n_{it} \longleftarrow 0$
tant que $modif$ et $n_{it} < N_{itMax}$ **faire**
  $\overline{*Etape\ 2*}$
  $\overline{(cc_{(k)})}$ //Ordonner les nuages par $card(cc_k)$ croissant
  $\overline{*Etape\ 3*}$
  pour $k = 1$ **à** C//Pour tous les nuages identifiés par Etape 1 **faire**
    $d_{intranuage}(k) \longleftarrow 0$
    pour $\forall a_i \in cc_k$ **faire**
      $d_{intranuage}(k) \longleftarrow d_{intranuage}(k) + min(d(a_i, a_j)), a_j \in cc_k, j \neq i$
    fin pour
    $d_{intranuage}(k) \longleftarrow \frac{d_{intranuage}(k)}{card(cc_k)}$
  fin pour
  modif $\longleftarrow$ FAUX
  $n_{it} \longleftarrow n_{it} + 1$
  $\overline{*Etape\ 4*}$
  pour $\forall cc_{(k)}$ /*nuage k*/ **faire**
    pour $\forall a_i \in cc_{(k)}$ /*point i*/ **faire**
      $\overline{**Etape\ 4a**}$
      $\overline{dmin \longleftarrow min\{d(a_i, a_j)\}, a_j \in cc_{(l)}, l > k}$ /*le point j appartient au nuage l*/
      $\overline{**Etape\ 4b**}$
      si $dmin < d_{intranuage}((k))\ ET\ dmin < d_{intranuage}((l))$ **alors**
        $label(a_i) \longleftarrow (l)$; $card(cc_{(k)}) - -$; $card(cc_{(l)}) + +$
        modif $\longleftarrow$ VRAI
      fin si
    fin pour
  fin pour
fin tant que

- Fusionner les résultats partiels pour prendre une décision sur l'appartenance finale des points à classifier.

Comme présenté dans les chapitres introductifs, un système de classification basé sur des règles est construit et utilisé en trois étapes :

1. L'étape d'apprentissage, où les règles sont déduites à partir d'un ensemble d'apprentissage.
2. L'étape de validation, où les règles sont appliquées sur des données dont on connaît la classe

d'appartenance, ce qui permet la validation de la méthode d'apprentissage, par exemple à l'aide d'un indice de bonne classification.

3. La dernière étape est l'étape d'exploitation du système réalisé en appliquant les règles sur des données inconnues ou qui n'ont pas servi à l'apprentissage.

Pour pouvoir étendre l'approche de base au cas multi–dimensionnel, il faut tenir compte d'une remarque très importante : les implications ne sont ni commutatives ni associatives. L'extension du principe à plus de deux attributs n'est donc pas immédiate. Passer du polygone convexe dans le plan à une forme linéaire tri-dimensionnelle par exemple ne peut pas être imaginé en gardant en même temps l'utilisation de l'opérateur d'implication dans la prémisse des règles.

La solution proposée pour la situation où les points à traiter ont plus de deux attributs est donc d'appliquer le principe décrit sur chaque paire d'attributs possible. Chaque paire d'attributs sera donc associée à une règle de classification. Il en résulte alors un autre problème : celui de fusionner les résultats issus de toutes ces règles afin d'obtenir un résultat final pertinent. Deux solutions possibles sont proposées et analysées dans les sections suivantes.

4.2.1 Méthodes de fusion des systèmes de classification

Comme la sortie des boîtes élémentaires qui ont été détaillées dans le chapitre précédent est une décision nette d'appartenance ou non–appartenance à la classe analysée, cette section est centrée sur les méthodes de fusion des systèmes de classification nette. Des méthodes proposées pour réaliser la fusion de classifieurs flous peuvent être trouvées dans la littérature [136, 56].

La fusion des classifieurs est d'habitude dictée par le besoin d'améliorer les résultats de la classification. Chaque classifieur qui est inclus dans le réseau de fusion apporte une certaine quantité d'information utile et le but du système de fusion est d'obtenir des résultats plus pertinents que chaque classifieur individuel.

Généralement, on peut identifier deux types d'approches dans le domaine :

1. la première consiste à évaluer les classifieurs individuellement et, en utilisant des critères pré–définis, en choisir un ou plusieurs qui sont considérés "les meilleurs" [157]. Si le principe est de choisir un seul classifieur, sa sortie est utilisée telle qu'elle est. Si la sélection consiste à choisir un groupe de classifieurs qui donnent des résultats pertinents, elle est d'habitude suivie par une deuxième étape qui consiste à appliquer une fusion selon la deuxième approche, décrite ci–après.
2. La deuxième approche consiste à prendre en compte tous les systèmes de classification analysés et calculer une sortie globale qui est une combinaison des sorties particulières de chaque classifieur [52, 158, 149, 140].

Dans le cadre de la première approche, la méthode la plus répandue est la méthode de sélection dynamique du classifieur. Cette méthode choisit un seul des classifieurs analysés et la sortie de ce classifieur est fournie comme sortie du système de fusion. Si le processus est répété en éliminant à chaque fois le classifieur qui a été considéré comme le meilleur dans l'étape précédente, on peut obtenir une hiérarchie des classifieurs selon leur pertinence [55, 157].

Une autre méthode appartenant à cette catégorie est la méthode de groupement et de structuration des classifieurs. Dans ce cas, comme illustré en [68], les classifieurs sont groupés selon des critères pré–définis. Différentes méthodes de sélection sont ensuite appliquées à chaque groupement afin de sélectionner le meilleur. Les critères, qui sont à la base de la construction des groupes et de la

sélection du "leader" de chaque groupe, sont très diverses et généralement dépendent de l'application considérée.

La deuxième approche peut être utilisée soit indépendamment sur l'ensemble de classifieurs à analyser, soit comme une deuxième étape de fusion sur le sous–ensemble de classifieurs fourni par une première étape de sélection. Dans ce cas, le résultat de chaque classifieur indépendant est pris en considération. Le système de fusion le plus fréquent est basé sur le principe du vote [138]. En général, les systèmes implémentés sur ce principe sont appliqués quand les classifieurs individuels fournissent une sortie nette et unique. La sortie du système de fusion est alors la classe qui apparaît le plus souvent dans les sorties des classifieurs individuels. Une condition supplémentaire est d'habitude rajoutée en ce qui concerne la fréquence absolue d'apparition de cette classe majoritaire. Ce critère est d'habitude un seuil qui donne la proportion minimale des votants qui doivent fournir le même résultat afin qu'il soit pris en considération. Ce seuil est souvent fixé à 0.5, ce qui est connu dans la littérature comme le "vote majoritaire".

Une autre méthode qui prend en considération tous les classifieurs de base analysés est une méthode qui organise les classes de sortie selon des rangs. Cette méthode est composée de deux étapes qui peuvent être aussi appliquées indépendamment. La première étape consiste à réduire le nombre de classes de sortie possibles. Le principe de cette étape est de réduire le plus possible l'ensemble des classes de sortie, en gardant en même temps une bonne probabilité d'inclusion de la vraie classe de sortie dans cet ensemble. Comme illustré en [68], dans le domaine on trouve deux grandes directions, l'union et l'intersection des voisinages. Ensuite la deuxième étape consiste à ordonner les classes qui se trouvent dans cet ensemble réduit de classes de sortie possibles. Le but de cette étape est de placer la vraie classe d'appartenance à la première place, c'est–à–dire lui associer la plus grande probabilité de sortie. Ce problème est aussi traité dans [68], les différentes approches pour le résoudre étant la méthode du plus grand rang, celle du compte de Borda, ou encore la régression logistique (méthode statistique qui permet de produire un modèle de prédiction des valeurs prises par une variable catégorielle à partir d'une série de variables explicatives).

Un problème particulier est de fusionner des classifieurs qui n'ont pas de sorties nettes, mais floues, c'est–à–dire dont les sorties sont des vecteurs composés par des valeurs réelles dans le domaine $[0, 1]$. Le but des méthodes de fusion dans ce cas–là est de réduire le degré d'incertitude associé aux degrés d'appartenance flous fournis par chaque classifieur individuel. Plusieurs méthodes peuvent être énumérées pour cette problématique.

Les méthodes de fusion basées sur le principe de Bayes peuvent être appliquées dans ce contexte si les sorties des classifieurs individuels sont exprimées comme des probabilités a posteriori des classes correspondantes, c'est–à–dire que la somme des composantes réelles du vecteur de sortie doit être égale à 1. La méthode basée sur la moyenne de Bayes consiste à calculer une probabilité a posteriori moyenne pour toutes les classes de sortie, par le simple calcul de la moyenne des composantes de même rang dans tous les vecteurs de sortie. Le principe est en fait une généralisation du principe de vote. La méthode peut être raffinée en prenant en compte les erreurs introduites par chaque classifieur et donc en leur associant des poids différents [90].

Une image d'ensemble sur les principales méthodes de fusion existantes peut être trouvée dans [136]. Une comparaison entre les performances de différentes stratégies de fusion est fournie dans [94] et dans [95]

Afin de fusionner les classifieurs de base décrits à la fin du chapitre précédent, on propose deux approches différentes. Comme principe de base, elles utilisent le principe de votes, qui a été adapté afin de traiter des sorties multiples qui ne sont pas des probabilités a posteriori. La différence principale entre les deux méthodes est dictée par l'importance associée au type d'information apportée par

chacune. La première approche apporte surtout de l'information "positive", elle est "permissive" avec l'appartenance des points à classifier aux classes apprises, alors que la deuxième apporte plutôt de l'information "négative", les points sont alloués à une classe selon des critères plus stricts. Les systèmes développés sont décrits dans les sections suivantes.

4.2.2 Approche basée sur les classes

Comme chaque paire d'attributs sera traitée indépendamment, une règle composée de plusieurs contraintes est calculée pour chaque paire et pour chaque classe pour laquelle on a des points d'apprentissage. Ces règles sont les éléments de base dans le système de classification proposé. Un premier mode d'utilisation de ces éléments de base est présenté dans la figure 4.6.

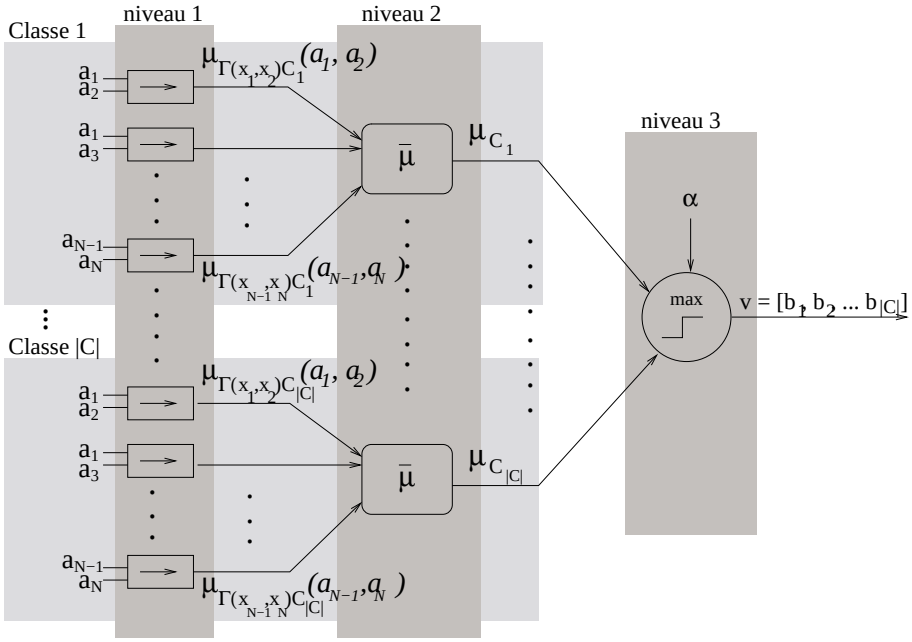


FIGURE 4.6 – Le système de classification - approche basée sur les classes

Cette approche est basée sur les classes, c'est–à–dire qu'elle va calculer un degré d'appartenance global μ_{C_i} pour chaque classe indépendamment les unes des autres (blocs horizontaux sur la figure 4.6).

Le traitement des informations est lui organisé sur trois niveaux (blocs verticaux sur la figure 4.6). Le premier niveau utilise chaque paire d'attributs du point à classifier afin de calculer son degré d'appartenance à une classe C_i. L'application d'une règle associée à la classe C_i et à un couple d'attributs $\{x_j, x_k\}$ constitue la "boîte élémentaire" du système de classification, comme décrit à la fin du chapitre précédent. La sortie d'une telle boîte élémentaire est un degré d'appartenance binaire du point à la classe analysée, comme exprimé dans l'équation (3.10), page 57.

Le deuxième niveau de traitement calcule un degré d'appartenance à chaque classe C_i, $\mu_{C_i}, i = \overline{1, |C|}$ selon un système de type "vote". Chaque paire d'attributs a le même poids dans le calcul de ce degré. Le résultat du vote μ_{C_i} sera alors le rapport entre le nombre de votants qui ont pris la décision "*le point appartient à la classe* C_i" sur le nombre total de votants, comme montré dans l'équation 4.4. Le nombre des votants est égal au nombre de combinaisons de deux attributs parmi N, noté par la suite $C_N^2 = \frac{N \cdot (N-1)}{2}$. Chaque degré d'appartenance μ_{C_i} a donc une valeur dans l'intervalle $[0, 1]$. Les extrémités de cet intervalle correspondent au vote unanime des paires d'attributs pour

l'appartenance ($\mu_{C_i} = 1$) ou la non–appartenance ($\mu_{C_i} = 0$) du point n–dimensionnel à la classe analysée.

$$\mu_{C_i} = \frac{\sum_{j=1}^{N-1} \sum_{k=j+1}^{N} \mu_{\Gamma_{(x_j,x_k)C_i}}(a_j, a_k)}{C_N^2} = \frac{2 \times \sum_{j=1}^{N-1} \sum_{k=j+1}^{N} \mu_{\Gamma_{(x_j,x_k)C_i}}(a_j, a_k)}{N(N-1)} \tag{4.4}$$

Le dernier niveau de traitement (niveau 3) réalise la fusion de tous ces degrés d'appartenance et retourne un vecteur binaire. Chaque composante de ce vecteur correspond à une classe et la valeur 0 indique la non–appartenance à cette classe, alors que la valeur 1 indique l'appartenance. Sur chaque composante du vecteur, la décision est prise par comparaison du résultat du vote à un seuil (noté α). Le choix de la valeur du seuil (entre 0 et 1) permet d'optimiser la proportion des exemples rejetés par rapport au degré de confiance dans la classification des exemples classifiés. L'interprétation de cette sortie sera détaillée dans une section suivante.

4.2.3 Approche basée sur les paires d'attributs

La deuxième approche se focalise sur un même couple d'attributs et non plus sur une classe recherchée. Cette approche est illustrée sur la figure 4.7. Cette fois–ci, les blocs horizontaux consistent à prendre une décision à partir d'un seul couple d'attributs.

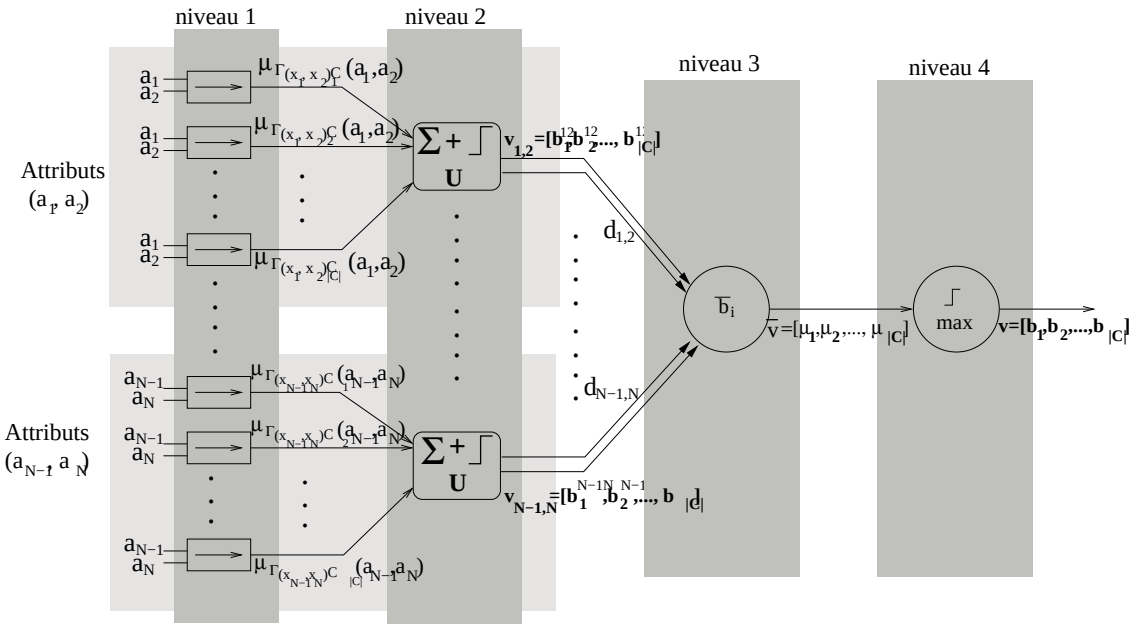


FIGURE 4.7 – Le système de classification - approche basée sur les paires d'attributs

Le flux de traitement va se découper en quatre niveaux qui sont légèrement différents de ceux de l'approche précédente.

Le premier niveau de traitement consiste à appliquer toutes les règles obtenues pour chaque classe à chacune des paires d'attributs disponibles. Ce niveau est identique pour cette approche et pour l'approche présentée dans la section précédente. Si on considère l'objet à classifier caractérisé par ses attributs $\{a_1, \ldots a_N\}$, alors la sortie de ce niveau est composée de plusieurs degrés partiels

d'appartenance donnés par chaque paire d'attributs (i, j) de l'objet analysé à chaque classe C_k : $\mu_{\Gamma(x_i,x_j)C_k}(a_i, a_j)$; $k \in \{1, \ldots, \mid C \mid\}$, $i \in \{1, \ldots, N-1\}$ et $j \in \{i, \ldots, N\}$.

Le deuxième niveau est l'étape la plus importante pour cette approche. C'est sur les aspects liés à cette étape que les deux méthodes sont profondément différentes. Plusieurs traitements sont effectués :

1. Somme des sorties des classifieurs élémentaires $k \in \{1, |C|\}$ pour chaque paire d'attributs : on calcule la somme des valeurs binaires obtenues au niveau 1 pour toutes les classes.
2. Seuillage appliqué sur la somme obtenue : une somme trop importante signifierait que le couple d'attributs apporte beaucoup d'ambiguïté dans la séparation des classes recherchées. Il est décidé de ne conserver que les couples d'attributs discriminants. Pour cela la somme doit être inférieure à un seuil ρ pour que la décision du couple soit conservée. La sortie de ce niveau consiste donc dans une valeur logique d_{ij} qui indique si le couple des attributs (i, j) a un pouvoir discriminant et qui peut avoir les valeurs "vrai" ou "faux" :

$$d_{i,j} = \begin{cases} VRAI, & si \sum_{k=1,|C|} \mu_{\Gamma(x_i,x_j)C_k} \leq \rho \\ FAUX, & sinon \end{cases}$$

   Le seuil ρ a été fixé à $\frac{|C|}{2}$ (valeur optimale obtenue experimentalement). Cela permet d'"éliminer" une paire d'attributs si elle vote positivement pour plus d'une moitié des classes recherchées. On note les paires retenues p_{r_i} et leur nombre P_r.
3. Construction du vecteur de sortie : si l'étape précédente n'arrête pas le processus, on concatène les résultats de base obtenus pour la paire d'attributs (i, j) dans un vecteur binaire $|C|$–dimensionnel v_{ij} :

$$v_{i,j} = [\mu_{\Gamma(x_i,x_j)C_1}, \cdots, \mu_{\Gamma(x_i,x_j)C_{|C|}}]$$

   Les composantes binaires b_k^{ij} de ce vecteur sont en fait le vote positif ou négatif de la paire d'attributs pour la classe correspondante C_k.

Les vecteurs de sortie de la deuxième étape sont transformés (niveau 3) en un seul vecteur en calculant une moyenne sur chaque composante des vecteurs obtenus pour les différentes paires d'attributs qui ont été retenues. On obtient alors un vecteur intermédiaire $\overline{v}$, qui est calculé selon la procédure 4.5. Si toutes les paires d'attributs donnent des résultats trop ambigus la sortie de cette étape est un vecteur nul dont la dimension est égale au nombre de classes apprises.

Début
  Etape 1. Initaliser $\overline{v} = [0, \cdots, 0]$.
  Etape 2. Pour toutes les classes C_k :
    2a) Pour toutes les paires d'attributs (i, j) :
      Si $d_{i,j}$: $\overline{v}[k] = \overline{v}[k] + v_{i,j}[k]$
    2b)$\overline{v}[k] = \frac{\overline{v}[k]}{P_r}$
Fin

Procédure 4.5 – Calcul de la sortie du niveau 3

Le dernier niveau (niveau 4) réalise la prise de décision sur chaque composante du vecteur $\overline{v}$ obtenu à l'étape précédente. La prise de décision réside en une comparaison à un seuil. Cette opération correspond en fait à remplacer dans le vecteur $\overline{v}$ les valeurs inférieures à un seuil par zéro, comme

montré dans l'équation 4.5. Le seuil donne le degré de flexibilité du système : un seuil bas correspond à un système qui accepte la classification d'un objet dans une classe même si le nombre des votes positifs est faible, alors qu'un seuil haut correspond à accepter de classer un objet seulement si le nombre des votes positifs est important. Pour la première situation, il est plus probable de classifier un pourcentage plus important des objets, mais la certitude d'avoir une classification correcte est basse. Dans la deuxième situation, il est plus probable d'avoir des objets qui restent non–classifiés, mais la certitude sur l'appartenance des objets qui sont classifiés augmente. Le seuil S proposé par le système développé est fixé à 0.5, ce qui se traduit par retenir une classe comme possible classe d'appartenance pour l'objet si au moins une moitié des paires des attributs disponibles ont voté positivement pour cette appartenance.

$$b_c^s = \begin{cases} \overline{v_c}, & \overline{v_c} \geq S \; ou \\ 0, & sinon \end{cases} \tag{4.5}$$

Le dernier pas de traitement est la binarisation du vecteur de sortie, comme présenté dans l'équation 4.6. Les composantes maximales et celles qui sont très proches de ces valeurs (une différence maximale de 5% est permise) sont gardées comme correspondant à des classes d'appartenance possibles. Comme dans le cas précédent, la sortie du système n'est pas forcément une et une seule classe d'appartenance, mais elle peut se situer dans un des cas décrits dans la section suivante.

$$b_c = \begin{cases} 1, & b_c^s \in (b_c^s max - 5\%, b_c^s max] \\ 0, & sinon \end{cases} \tag{4.6}$$

Cette deuxième approche de fusion des différentes boîtes de base du système de classification peut être vue comme une post–sélection des attributs. Même si tous les attributs disponibles sont utilisés pendant l'étape d'apprentissage du système, il est possible de ne pas utiliser une partie de ces attributs (plus exactement une partie des paires d'attributs) pendant la classification elle–même, et cela selon le point à classifier. Le principe qui se trouve à la base de cette approche est le fait qu'une paire d'attributs peut être très discriminante et pertinente pour un point donné mais plutôt ambigüe pour un autre. D'un point de vue géométrique on peut analyser cette approche en se rapportant à la signification d'une boîte de base, telle qu'elle a été décrite dans le chapitre précédent. Si on prend en considération un même point et les différents plans de représentation donnés par différentes paires d'attributs, il peut se trouver en plein centre d'un polygone associé à une classe et seulement dans ce polygone pour certaines paires et en périphérie de plusieurs polygones qui se chevauchent pour d'autres. Le principe qui est à la base de l'approche décrite dans cette section est donc d'ignorer ces dernières paires d'attributs et de prendre en considération seulement les paires d'attributs qui sont vraiment discriminantes. Même si de cette manière on perd apparemment de l'information, le résultat final peut être plus fiable, comme cela est illustré ultérieurement à travers des benchmarks et des applications.

4.3 Interprétation de la sortie du système

Les deux approches proposées ont comme sortie un vecteur binaire. Le système accepte la situation d'ambiguïté et celle de rejet. L'ambiguïté correspond en fait à l'appartenance d'un même point à plusieurs classes, alors qu'un point rejeté est considéré comme étant en dehors de toutes les classes qui ont été apprises. Pour gérer toutes les possibilités, la sortie du système de classification proposé n'est donc pas une classe, mais un vecteur de degrés d'appartenance binaires $v = [b_1, b_2, \ldots, b_{|C|}]$,

comme suggéré dans les deux sections précédentes. Chacun de ces degrés donne l'appartenance ou la non–appartenance à une des classes existantes. On peut donc identifier plusieurs situations possibles :

- La classification nette : dans le vecteur de sortie il existe une seule valeur unitaire, c'est-à-dire $b_k = 1$; $\sum_{i=1}^{|C|} b_i = 1$, donc $b_k = 1$ et $b_i = 0\ \forall i \neq k$. Dans cette situation le point est classé directement dans la classe k.
- Le rejet : le vecteur de sortie n'est composé que de valeurs nulles ($\sum_{i=1}^{|C|} b_i = 0$). Cette situation est la situation de rejet, où le point n'est alloué à aucune des classes apprises. Elle correspond généralement à une quantité d'information "négative" importante.
- L'ambiguïté : dans le vecteur de sortie, il existe plusieurs valeurs unitaires ($\sum_{i=1}^{|C|} b_i > 1$). Cette situation est une situation d'ambiguïté. Elle est gérée en gardant l'ambiguïté et en créant une nouvelle classe associée à cette situation. Une ambiguïté importante correspond à une quantité d'information "positive" importante. On peut envisager de traiter dans un deuxième temps ces situations d'ambiguïté avec par exemple des connaissances supplémentaires. Ces travaux ne traitent pas cet aspect et se contentent de classer les points en classe d'ambiguïté.

4.4 Validation sur des benchmarks

Afin de valider le système proposé, il a été appliqué sur des bases de données classiques, usuellement utilisées comme références dans le domaine de la classification. Ces bases de données appelées "benchmarks" sont disponibles sur internet :

- http ://archive.ics.uci.edu/ml/datasets.html

Les résultats présentés concernent les benchmarks "iris" et "wine". Ces deux ensembles de données ont été choisis parce qu'ils sont très utilisés dans le domaine de la classification, ce qui permet de se placer par rapport aux autres classifieurs. Sachant que les règles floues conjonctives, qui sont la méthode de classification la plus proche de la méthode proposée, sont généralement acceptées comme règles puissantes de classification, une comparaison entre les résultats des deux méthodes est proposée.

La validation de la méthode a été faite en deux étapes principales. La première étape consiste à apprendre le système sur la totalité de l'ensemble de données disponibles et ensuite de la tester sur ces mêmes données. Le but de cette étape est de valider le système à un niveau de base et ainsi de montrer que l'utilisation des règles graduelles comme règles de classification ne donne pas de résultats aberrants.

Comme le système de classification proposé permet des sorties multiples (l'ambiguïté) et l'absence de sorties (le rejet), un simple taux de classification correcte n'est pas très pertinent. Les résultats seront donc présentés sous la forme d'une matrice de confusion légèrement modifiée par rapport à sa forme classique.

La deuxième étape de validation consiste à évaluer le système dans un cadre formel. Pour cela, un système d'évaluation basé sur le principe de cross–validation, notamment sur la méthode "leave–one–out" décrite dans la section 1.3.2, a été mis en œuvre.

4.4.1 Validation sur les données d'apprentissage "iris"

Le premier benchmark utilisé est l'ensemble de données "iris", qui est caractérisé par :

- 3 classes représentées chacune par 50 points (un total de 150 points).
- Chaque point est caractérisé par 4 attributs numériques avec des valeurs réelles, représentant la longueur et la largeur des sépales et des pétales.

Les tableaux 4.6 et 4.7 représentent les matrices de confusion obtenues en appliquant le système de classification proposé en configuration "approche par classes" (tableau 4.6) et "approche par paires d'attributs" (tableau 4.7). Les colonnes "C_0" à "C_2" représentent les pourcentages de points qui ont été alloués à la classe correspondante (et seulement à cette classe), la colonne "Confusion" le pourcentage de points qui ont été alloués à 2 classes, "Rejet" le pourcentage des points qui n'ont été alloués à aucune classe et "Ambig. totale" le pourcentage des points qui ont été alloués à toutes les classes disponibles. Sur les lignes sont représentées les points appartenant aux trois classes pour les quatre niveaux de pré–traitement proposés : sans pré–traitement (–), épuration de l'ensemble d'apprentissage (1), identification des composantes connexes (2) et épuration de l'ensemble d'apprentissage suivie par l'identification des composantes connexes (1 et 2).

On remarque dans les tableaux 4.6 et 4.7 que les résultats sont très bons. On peut également remarquer le comportement différent des deux approches : l'ambiguïté qui apparaît pour l'approche par classes se transforme en rejet pour l'approche par paires d'attributs, ce qui correspond à nos prévisions, étant donné les principes des deux approches : l'ambiguïté peut apparaître à cause de différentes paires d'attributs qui votent pour des classes différentes, mais plus souvent, si le système est bien appris et l'ensemble d'apprentissage cohérent avec l'ensemble de test, l'ambiguïté se retrouve au niveau de chaque paire d'attributs ; ainsi, une même paire va voter pour plusieurs classes. Dans le cas de l'approche par classes le vote sera pris en considération pour toutes ces classes, ce qui amène très probablement à propager l'ambiguïté jusqu'à la sortie finale du système. Par contre, pour l'approche par paires d'attributs ces paires qui apportent de l'ambiguïté dans le système sont tout simplement ignorées. Si la totalité ou la grande majorité des paires se trouvent dans cette situation, le système, en manque d'information pertinente, prend la décision de rejet.

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	100	0	0	0	0	0
	C_1	0	98	0	2	0	0
	C_2	0	0	100	0	0	0
1	C_0	100	0	0	0	0	0
	C_1	0	98	2	0	0	0
	C_2	0	0	100	0	0	0
2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
1 et 2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0

TABLE 4.6 – Premiers résultats obtenus sur l'ensemble de données "iris" – approche par classes

Ces résultats de base ont confirmé la possibilité d'utiliser les règles graduelles comme règles de classification. Ils sont aussi en parfaite cohérence avec les attentes théoriques déduites du modèle développé.

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	100	0	0	0	0	0
	C_1	0	98	0	0	2	0
	C_2	0	0	100	0	0	0
1	C_0	100	0	0	0	0	0
	C_1	0	98	2	0	0	0
	C_2	0	0	100	0	0	0
2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
1 et 2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0

TABLE 4.7 – Premiers résultats obtenus sur l'ensemble de données "iris" – approche par paires d'attributs

4.4.2 Validation sur les données d'apprentissage "wine"

Un deuxième test est réalisé sur l'ensemble de données "wine". Son but est de touver l'origine de différents vins à partir de résultats d'analyses chimiques. Cet ensemble de données est considéré comme très approprié pour le test de nouveaux systèmes de classification, même si la difficulté de la tâche de classification n'est pas trop élevée. Il comporte les caractéristiques suivantes :

- 3 classes représentées respectivement par 59, 71 et 48 points (un total de 178 points).
- Chaque point est caractérisé par 13 attributs numériques à valeur réelle, comme par exemple le pourcentage d'alcool, l'intensité de la couleur, l'alcalinité etc.

On peut remarquer que les premiers résultats, présentés dans les tableaux 4.8 et 4.9, sont très encourageants : seulement un point de la première classe a été placé dans la classe de rejet (il n'a été affecté à aucune des classes apprises) dans le cas de l'approche par paires d'attributs.

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
1	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
1 et 2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0

TABLE 4.8 – Premiers résultats obtenus sur l'ensemble de données "wine" – approche par classes

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	98.3	0	0	0	1.7	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
1	C_0	98.3	0	0	0	1.7	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0
1 et 2	C_0	100	0	0	0	0	0
	C_1	0	100	0	0	0	0
	C_2	0	0	100	0	0	0

TABLE 4.9 – Premiers résultats obtenus sur l'ensemble de données "wine" – approche par paires d'attributs

4.4.3 Test en généralisation sur les données "iris"

Dans les tableaux 4.10 et 4.11 sont présentés les résultats obtenus en cross–validation par la méthode "leave one out". On remarque une baisse importante de performance par rapport à la validation sur les données d'apprentissage, surtout pour l'approche par classes. Pourtant, les taux moyens de classification correcte restent au–dessus de 80% (90.67% pour l'apprentissage sans traitement de l'ensemble d'apprentissage, 88% pour l'ensemble d'apprentissage filtré, 85.33% quand l'algorithme d'identification des composantes connexes est appliqué sur l'ensemble d'apprentissage et 81.33% quand les deux traitements sont appliqués) pour l'approche par classes et au–dessus de 90% (respectivement 95.33%, 95.33%, 94% et 94%) pour l'approche par paires d'attributs. Ces valeurs sont calculées en faisant la moyenne sur les 3 taux de classification correcte rapportés dans les tableaux 4.10 et 4.11. Si on se réfère aux résultats rapportés pour différentes approches basées sur des règles conjonctives [75], où la méthode leave–one–out est aussi utilisée, on remarque que les taux de classification correcte varient entre 88.7% et 98%, ce qui place le classifieur proposé dans une position correcte par rapport aux classifieurs basés sur des règles conjonctives.

Le taux de rejet obtenu par le système proposé se situe pour l'approche par classes entre 5.33% et 14.67% (5.33%, 8%, 10.67%, 14.67%) et entre 0.67% et 1.33% pour l'approche par paires d'attributs (0.67%, 0.67%, 1.33% et 1.33%). Ces valeurs sont calculées de la même manière que les taux moyens de classification correcte. Les taux de rejet rapportés en [75] varient entre 0% et 10%, ce qui place encore une fois le système proposé dans une position acceptable. Afin d'évaluer le niveau d'erreur obtenu, on considère la confusion entre deux classes (dont l'une la classe réelle) comme une erreur de classification, même si cette décision contient des informations utiles. Dans ce contexte, on obtient les niveaux d'erreur de 4% pour l'approche par classes pour tous les niveaux de pré–traitement et de (4.67%, 4%, 4.67% et 4.67%) pour l'approche par paires d'attributs, alors que les niveaux d'erreur rapportés varient entre 0% et 8%.

On remarque l'influence plutôt négative de l'application des algorithmes de traitement. La raison principale de cette baisse de performances est le fait que ces algorithmes ont été conçus pour le cas des ensembles d'apprentissage de grande taille, où les mesures statistiques que l'on utilise ont une vraie raison d'être. Pour l'ensemble de données "iris", on dispose de 50 points d'apprentissage pour chaque classe, ce qui rend les résultats d'une analyse statistique peu pertinents.

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	90	0	0	0	10	0
	C_1	0	92	6	2	0	0
	C_2	0	2	90	2	6	0
1	C_0	84	0	0	0	16	0
	C_1	0	90	8	0	2	0
	C_2	0	2	90	2	6	0
2	C_0	74	0	0	0	26	0
	C_1	0	92	6	2	0	0
	C_2	0	2	90	2	6	0
1 et 2	C_0	64	0	0	0	36	0
	C_1	0	90	8	0	2	0
	C_2	0	2	90	2	6	0

TABLE 4.10 – Leave–one–out : matrice de confusion obtenue sur l'ensemble de données "iris" – approche par classes

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	100	0	0	0	0	0
	C_1	0	92	6	2	2	0
	C_2	0	2	94	4	0	0
1	C_0	100	0	0	0	0	0
	C_1	0	92	8	0	0	0
	C_2	0	2	94	2	2	0
2	C_0	96	0	0	0	4	0
	C_1	0	92	6	2	0	0
	C_2	0	2	94	4	0	0
1 et 2	C_0	96	0	0	0	4	0
	C_1	0	92	8	0	0	0
	C_2	0	2	94	4	0	0

TABLE 4.11 – Leave–one–out : matrice de confusion obtenue sur l'ensemble de données "iris" – approche par paires d'attributs

La différence importante de performances entre les résultats obtenus quand l'apprentissage est réalisé sur la totalité de l'ensemble de données et quand la méthode de leave–one–out est appliquée se justifie par le principe même du système de classification. Pour chaque paire d'attributs, chaque classe est associée à un polygone qui inclut au plus près tous les points d'apprentissage utilisés, et donc qui est dicté par les points–extrêmes de l'ensemble. L'élimination d'un de ces points–extrêmes de l'ensemble d'apprentissage provoque la restriction du polygone et donc l'application des règles obtenues sur ce point ne va pas l'identifier comme appartenant à la classe étudiée, comme montré dans la figure 4.8. Cette figure représente les points d'apprentissage pour l'ensemble de données "iris" pour la première classe. Quand le système apprend les règles sur l'ensemble d'apprentissage qui ne contient pas le point–extrême figuré les règles obtenues ne couvrent pas (pour le couple d'attributs figuré) le point qui a été éliminé. Pourtant, cet effet peut ne pas être visible si le nombre d'attributs est assez important, étant donné qu'un point–extrême dans l'espace associé à une paire d'attributs n'est pas forcément un point–extrême pour les autres paires.

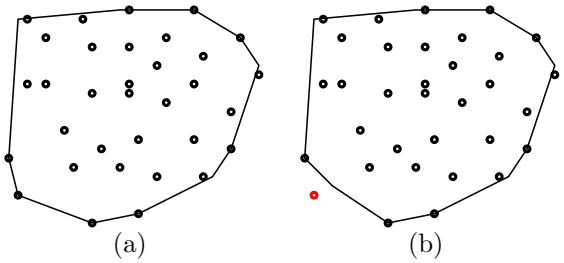


FIGURE 4.8 – L'effet de l'élimination d'un point d'apprentissage extrême pour une paire d'attributs et une classe de l'ensemble iris

4.4.4 Test en généralisation sur les données "wine"

Les résultats obtenus sur l'ensemble de données "wine" sont présentés dans les tableaux 4.12 et 4.13. Les taux de classification correcte varient donc entre 83.86% et 88.38% pour l'approche par classes (**88.38%**, **83.86%**, 87.44% et 87.44%) et entre 86.05% et 91.13% pour l'approche par paires d'attributs (**91.13%**, **86.05%**, 86.94% et 86.94%). Les résultats rapportés dans [79], qui utilisent la même méthode d'évaluation, indiquent des taux de classification correcte situés entre 85.96% et 95.51%. Les résultats obtenus avec la méthode proposée se situent une nouvelle fois dans le même ordre de grandeur que ceux provenant des systèmes de classification basés sur des règles floues conjonctives.

La remarque concernant l'applicabilité des algorithmes de pré–traitement reste valabale. Malheureusement, on ne dispose pour cet ensemble de données que de 48 à 71 points pour chaque classe, ce qui ne permet pas une analyse statistique pertinente.

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	76.28	18.64	0	5.08	0	0
	C_1	0	97.18	0	0	2.82	0
	C_2	0	2.08	91.67	2.08	4.17	0
1	C_0	62.72	33.9	0	1.69	1.69	0
	C_1	0	97.18	0	0	2.82	0
	C_2	0	2.08	91.67	2.08	4.17	0
2	C_0	76.28	16.95	0	0	6.77	0
	C_1	0	94.37	0	0	5.63	0
	C_2	0	4.17	91.67	0	4.17	0
1 et 2	C_0	76.28	16.95	0	0	6.77	0
	C_1	0	94.37	0	0	5.63	0
	C_2	0	2.08	91.67	2.08	4.17	0

TABLE 4.12 – Leave–one–out : matrice de confusion obtenue sur l'ensemble de données "wine" – approche par classes

Pré-traitement		C_0[%]	C_1[%]	C_2[%]	Confusion[%]	Rejet[%]	Ambig. totale[%]
–	C_0	79.66	15.25	0	1.69	3.39	0
	C_1	0	100	0	0	0	0
	C_2	0	2.08	93.75	0	4.17	0
1	C_0	64.41	16.95	0	0	18.64	0
	C_1	0	100	0	0	0	0
	C_2	0	6.25	93.75	0	0	0
2	C_0	67.8	5.08	0	0	27.12	0
	C_1	0	97.18	0	0	2.82	0
	C_2	0	2.08	95.84	0	2.08	0
1 et 2	C_0	67.79	8.47	0	0	23.74	0
	C_1	0	97.18	0	0	2.82	0
	C_2	0	2.08	95.84	0	2.08	0

TABLE 4.13 – Leave–one–out : matrice de confusion obtenue sur l'ensemble de données "wine" – approche par paires d'attributs

4.5 Conclusion sur le système obtenu

Ces résultats ont encouragé l'utilisation de la méthode de classification proposée sur des applications réelles, qui seront présentées dans le chapitre suivant.

Les deux approches proposées pour l'utilisation du système d'apprentissage proposé dans le chapitre précédent sont complémentaires. L'approche par classes permet aux paires d'attributs de voter pour plusieurs classes, ce qui peut amener à des ambiguïtés dans la sortie. L'avantage de cette approche est que le taux de rejet sera très faible et donc pour les applications où la sensitivité est importante, elle est une approche intéressante (elle produit un faible taux de résultats "false negative").

L'approche par paires d'attributs interdit l'ambiguïté au niveau de chaque couple d'attributs. Cela diminue la probabilité d'avoir l'ambiguïté dans la sortie finale, mais en même temps augmente la probabilité du rejet comme décision finale. En conséquence, elle est préférable pour les applications où la spécificité et la précision sont importantes (le taux de "false positive" est minimisé).

Chapitre 5

Applications

Le système de classification proposé a été appliqué dans le cadre de deux applications. La première est une application industrielle qui concerne l'analyse de pièces électroniques à l'aide d'images tomographiques 3D. La deuxième est une application dans le domaine de l'imagerie satellitaire radar qui consiste à identifier différentes zones spécifiques.

5.1 Analyse d'images tomographiques 3D

5.1.1 Problématique

Les pièces étudiées dans ces travaux sont fabriquées en matériaux composites, un mélange principalement composé d'une résine polymère (appelée matrice) et de fibres de verre, comme celle montrée sur la figure 5.1. Elles sont conçues par la société Schneider Electric, un des leaders mondiaux en appareillages électrotechniques.

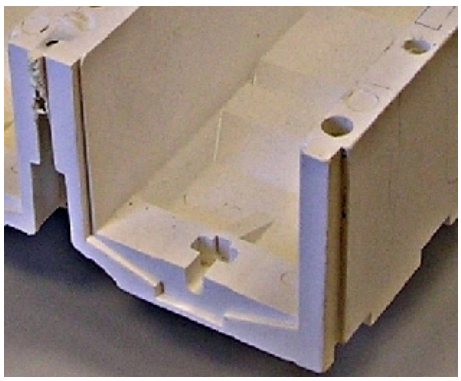


FIGURE 5.1 – Exemple d'une pièce électrotechnique en matériau composite

Il s'agit de pièces isolantes contenues dans des disjoncteurs basse et moyenne tension. L'organisation des fibres de verre et de la résine au sein du matériau influe sur les performances diélectriques et thermo-mécaniques de la pièce et donc sur la qualité de celle-ci. Les premières études menées pour visualiser l'organisation du matériau consistaient à découper les pièces puis à les analyser au microscope, ce qui les endommageait définitivement. De plus, le découpage détériorait l'intérieur du matériau et donnait alors une image biaisée de l'organisation des fibres et de la résine. Schneider Elec-

tric s'est alors tourné vers une méthode non destructive pour analyser les pièces, avec la tomographie à rayons X.

Un faisceau de rayons X est projeté au travers de la pièce (cf. figure 5.2). Un récepteur recueille les rayons l'ayant traversée afin de délivrer une image en deux dimensions de l'intérieur de la pièce. Cette image, appelée radiographie, informe sur l'intégralité du contenu de la pièce tout au long du trajet des rayons.

La tomographie tridimensionnelle (technique proche des scanners médicaux) permet d'obtenir des informations détaillées sur l'organisation interne du matériau. Elle consiste tout d'abord à imager plusieurs fois la pièce en la faisant tourner sur elle-même. Puis, cette série d'images est traitée par des logiciels permettant la reconstruction d'une image tomographique en 3D de la pièce [25, 71].

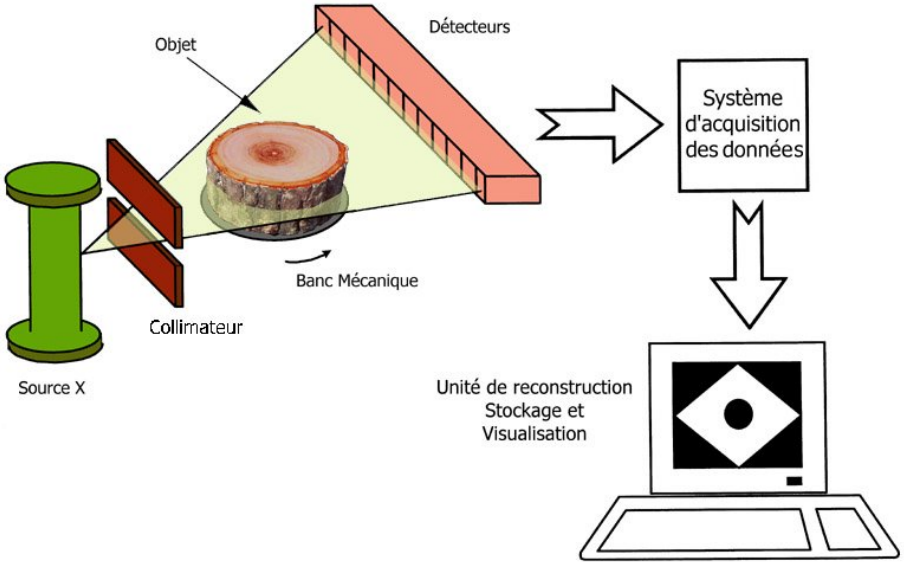


FIGURE 5.2 – Schéma de principe de la tomographie à rayon X.

Les tomographies tridimensionnelles mises à notre disposition sont des images (ensemble de voxels) en niveaux de gris codés sur 8 bits non signés. La figure 5.3 présente une image tomographique qui est étudiée dans ce document. Les niveaux de gris retranscrivent la densité des différents composants contenus dans le matériau composite. Ainsi, dans ces images, les fibres de verre (ou faisceaux de fibres) apparaissent en blanc. La résine utilisée dans la confection des pièces va elle ressortir sous la forme d'une texture fine avec un niveau de gris moyen. Enfin, les zones noires sont des trous, c'est-à-dire des zones où il n'y a pas de matière [126].

Actuellement, les personnes de la société Schneider Electric, chargées de l'expertise et du contrôle de la qualité des pièces en matériaux composites, analysent les blocs d'images tomographiques, section par section, pour déterminer la position et le volume de régions ayant des propriétés physiques spécifiques [147]. L'emplacement et la taille de ces régions ont un impact direct sur la qualité des pièces en matériaux composites. Mais cette analyse demande beaucoup de temps et de connaissances pour retrouver toutes les régions intéressantes.

Pour cette application, l'objectif est de mettre en œuvre un système d'aide à l'interprétation d'images tomographiques 3D par la classification des différentes régions présentes dans les images. Cette classification est fonction de l'organisation particulière des fibres de verre et de la résine et elle aidera les experts à mieux comprendre le contenu des images tomographiques. Trois types de régions sont recherchés dans cette étude. Ces régions, illustrées dans la figure 5.4 se décrivent de la façon

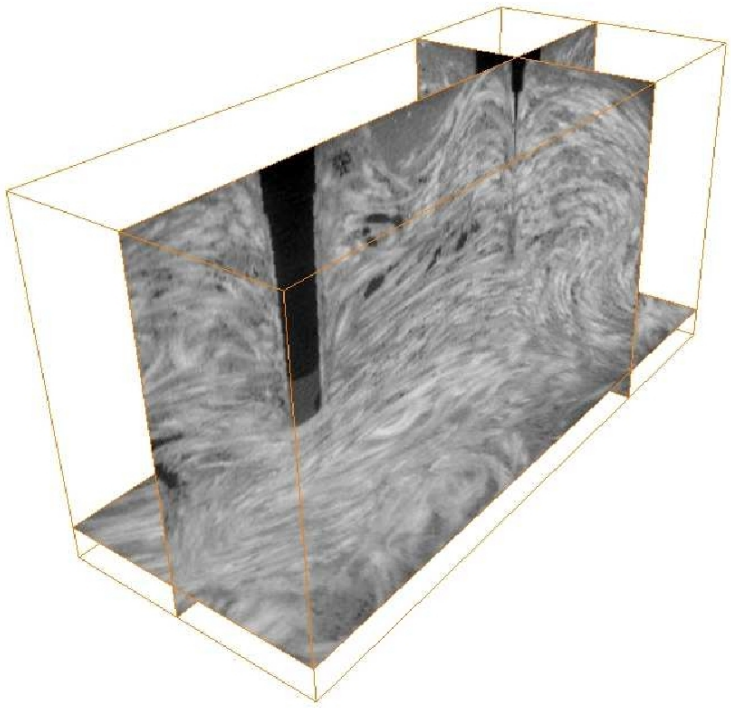


FIGURE 5.3 – L'image tomographique 3D étudiée (271x522x215 voxels)

suivante :

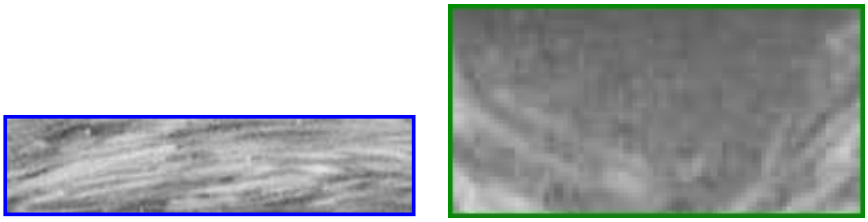
- Régions orientées (figure 5.4(a)) : régions contenant des fibres de verre orientées dans une même direction.
- Régions non–orientées (figure 5.4(c)) : régions contenant des fibres de verre qui s'enchevêtrent dans des directions aléatoires.
- Régions de manque de renfort (figure 5.4(b)) : régions principalement composées de résine avec très peu de fibres de verre.

Des travaux précédents [111] ont porté sur l'analyse de ces données au moyen de techniques de traitement d'image. Des attributs ont ainsi été mis au point pour détecter des caractéristiques propres à chaque région d'intérêt :

- Pour les régions orientées et non orientées, des attributs dédiés à la quantification de l'organisation des fibres dans l'image
- Pour les manques de renfort, des attributs spécialisés dans la caractérisation des textures

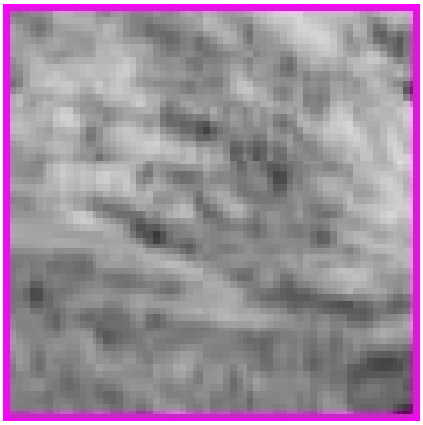
Dans les descriptions des attributs qui vont suivre, des régions typiques sont proposées en illustration ainsi que la partie de l'image tomographique qui leur est associée.

Le premier type d'attributs caractérise l'organisation des niveaux de gris. Les attributs de cette catégorie sont notés A_1, A_2 et A_3. La caractérisation des orientations au sein des images tomographiques a été effectuée avec une mesure de l'orientation des variations de niveau de gris basée sur les gradients [39]. Cette approche est liée à la fréquence des alternances entre les faibles et les forts niveaux de gris. Plusieurs mesures sont ensuite employées pour caractériser ces alternances. Trois d'entre elles ont été appliquées et ont permis d'obtenir les attributs A_1, A_2 et A_3. Dans ces attributs, le niveau de gris de chaque voxel de l'image correspond à un degré d'orientation des fibres dans l'image tomographique, comme l'illustre la figure 5.5.

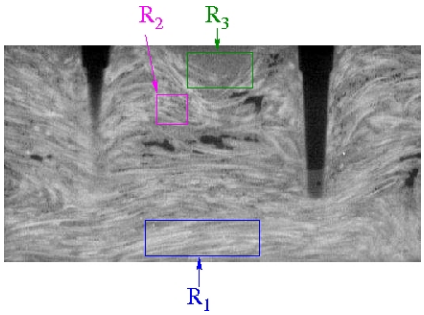


(a) Région orientée

(b) Région manque de renfort

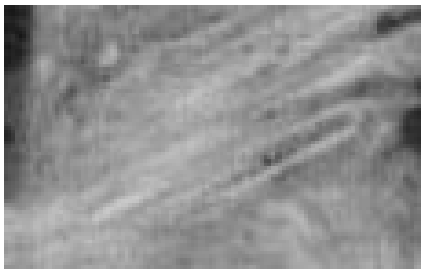


(c) Région non–orientée

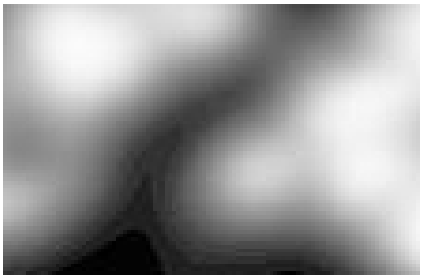


(d) Zones d'interêt sur la pièce étudiée

FIGURE 5.4 – Régions typiquement recherchées



(a) Image originale



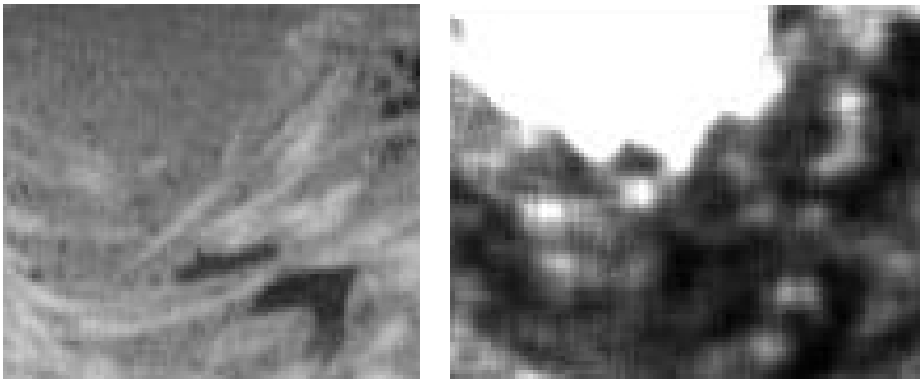
(b) Exemple d'attribut obtenu

FIGURE 5.5 – Attributs qui caractérisent l'organisation des niveaux de gris – pixels claires correspondent aux régions non–orientées, les pixels sombres correspondent aux régions orientées

Le deuxième type d'attributs caractérise les textures de l'image. Un seul attribut de ce type a été calculé, il est noté A_4. La caractérisation des textures homogènes s'appuie sur la matrice de cooccurence [65]. Cette matrice permet d'acquérir différentes mesures définissant les textures de l'image en modélisant les variations de niveaux de gris dans le voisinage d'un voxel donné. A partir des informations contenues dans cette matrice, une mesure d'homogénéité est calculée en chaque voxel. Un exemple est présenté dans l'image 5.6.

5.1.2 Résultats

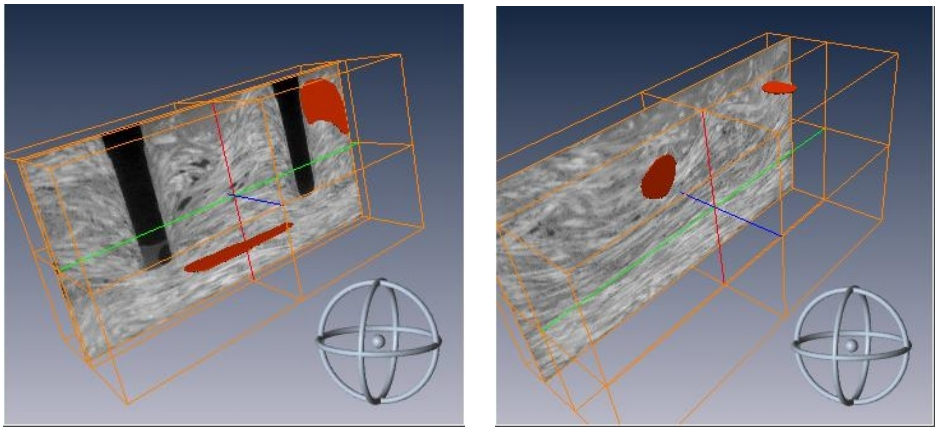
Le système de classification proposé précédemment a été appliqué sur les 4 attributs A_1, A_2, A_3, A_4. Un ensemble d'apprentissage a été construit à l'aide d'expertise : sur les images 3D, les experts ont



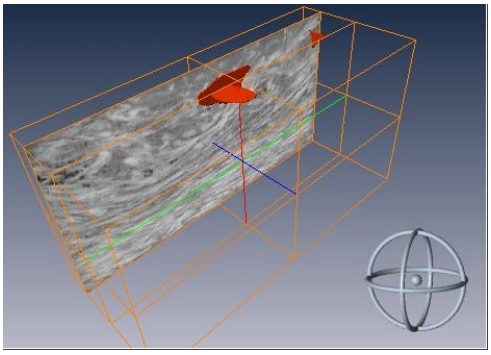
(a) Image originale (b) Attribut obtenu

FIGURE 5.6 – Attribut qui caractérise les textures – les pixels clairs indiquent la présence d'une texture

détouré des zones typiques, connues comme appartenant à une des classes recherchées. Les régions pointées sont représentées dans les figures 5.7(a), 5.7(b) et 5.7(c). Les vecteurs 4–dimensionnels des attributs caractérisant les pixels situés à l'intérieur de ces zones pointées ont servi comme ensemble d'apprentissage.



(a) Pointés de référence – classe orientée (b) Pointés de référence – classe non–orientée



(c) Pointés de référence – classe manque de renfort

FIGURE 5.7 – Pointés de référence

La validation du système a été faite de deux manières différentes. La première est une évaluation

qualitative. Elle consiste à appliquer les règles obtenues à partir des ensembles d'apprentissage sur l'intégralité des images disponibles, ce qui offre aux experts la possibilité d'une évaluation visuelle des performances. La deuxième validation est quantitative et elle s'appuie sur des pointés spécifiques réalisés par les experts (représentées dans la figure 5.8). Ces pointés ne servent pas à l'apprentissage mais uniquement à la généralisation.

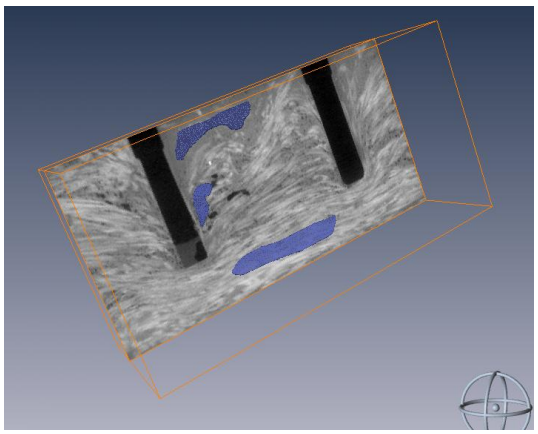
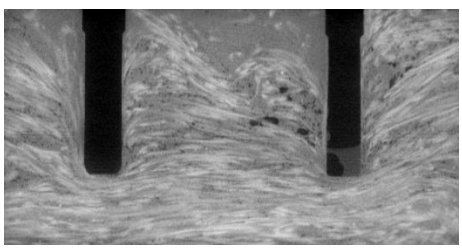


FIGURE 5.8 – Pointés de test

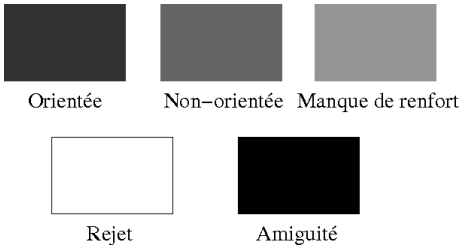
Les images présentées dans la figure 5.9, représentant des sections 2D provenant des images 3D, présentent la classification obtenue pour les deux approches (par classe et par paires d'attributs). Elles ont été sélectionnées afin d'illustrer les points forts et les points faibles du système de classification. La figure 5.9(a) présente une section verticale de la pièce 3D analysée. On peut remarquer la présence des zones orientées dans la partie inférieure, des zones non-orientées au centre de l'image et des zones de manque de renfort dans la partie supérieure de la pièce. L'analyse qualitative de ces résultats révèle une bonne détection des zones orientées et des manques de renfort pour les deux approches et pour les 4 niveaux de pré–traitement. Par contre, la détection des zones non–orientées est assez faible, à l'exception de l'approche par classe en apprenant sur l'ensemble d'apprentissage non–traité. Les autres résultats placent les régions dans l'ambiguïté (approche par classe) et dans le rejet (approche par paires d'attributs). Il est intéressant de remarquer que l'application soit de l'épuration des points non–pertinents, soit de l'identification des composantes connexes n'apporte pas beaucoup d'informations. Par contre, leur application "en cascade" réduit significativement les régions placées en ambiguïté par l'approche par classes et en rejet par l'approche par paire d'attributs et mène à une meilleure identification des zones non–orientées (figure 5.9(j)).

Une autre section, cette fois dans le plan xz, est présentée dans la figure 5.10(a). Encore une fois, de façon qualitative, on remarque une détection plutôt bonne des zones de manque de renfort (haut de la pièce) et orientée (la partie inférieure de la pièce, à l'exception du coin inférieur gauche, où les fibres sont plutôt non–orientées). Par contre, les régions non–orientées (la partie supérieure–centrale, côté droit) ne sont pas très clairement identifiées, sauf quand on applique "en cascade" les deux pré–traitements. Ces résultats ont confirmé l'utilité des deux niveaux de pré–traitement.

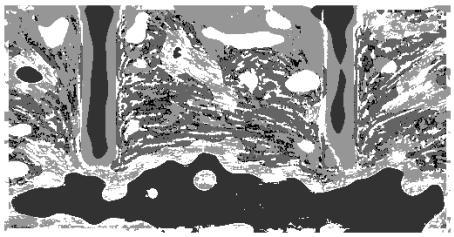
Comme première validation quantitative du système de classification, la méthode leave–one–out a été appliquée. Les résultats sont présentés dans les tableaux 5.1 (pour l'approche par classes) et 5.2 (pour l'approche par paires d'attributs). La forme particulière de ces matrices de confusion nécessite quelques précisions. Comme le système de classification développé permet d'avoir en sortie un degré d'appartenance unitaire à plusieurs classes, ce cas d'ambiguïté est présenté dans les colonnes 5 et 6 : "l'ambiguïté positive" signifie l'ambiguïté entre plusieurs classes à condition que la classe réelle fasse partie de cet ensemble, alors que "l'ambiguïté négative" signifie l'ambiguïté entre les deux autres



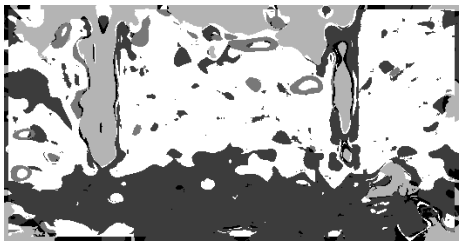
(a) Image originale, séction xy



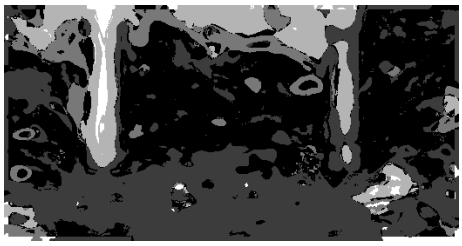
(b) Légende



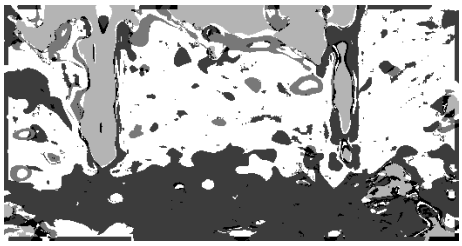
(c) Ensemble d'apprentissage non traité, approche par classes



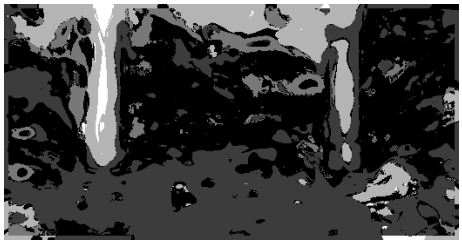
(d) Ensemble d'apprentissage non traité, approche par paires d'attributs



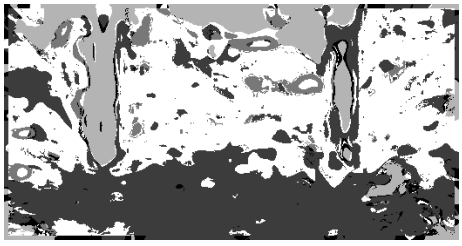
(e) Ensemble d'apprentissage filtré, approche par classes



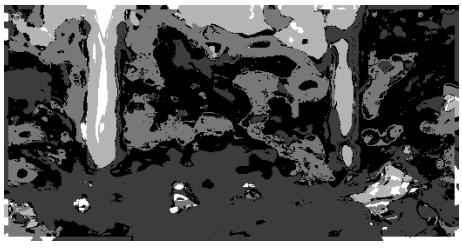
(f) Ensemble d'apprentissage filtré, approche par paires d'attributs



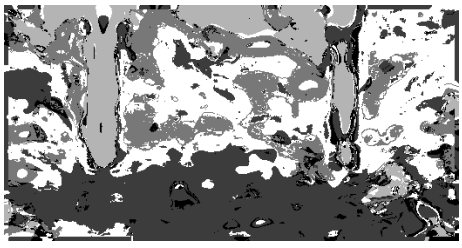
(g) Identification composantes connexes, approche par classes



(h) Identification composantes connexes, approche par paires d'attributs



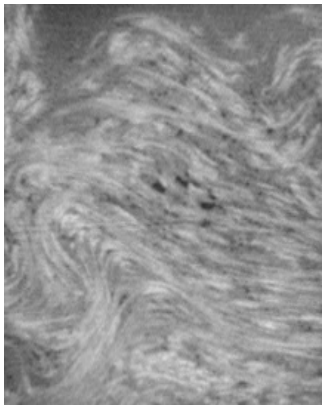
(i) Les deux traitements, approche par classes



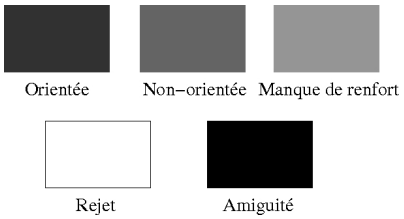
(j) Les deux traitements, approche par paires d'attributs

FIGURE 5.9 – Classification obtenue pour l'application Schneider, section verticale

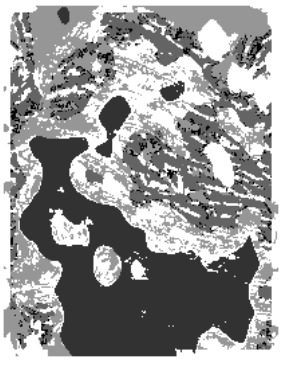
classes qui ne contiennent pas la classe réelle. Par exemple, si un point à classifier appartient à la classe "régions orientées" et s'il est affecté en même temps aux classes "régions orientées" et "régions manque de renfort", il sera comptabilisé dans le cas "ambiguïté positive". Si par contre il est affecté



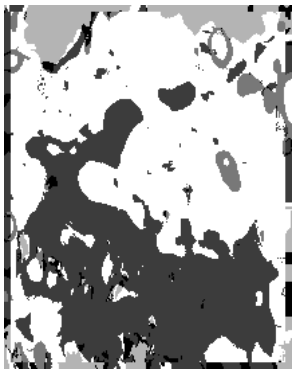
(a) Image originale, séction xz



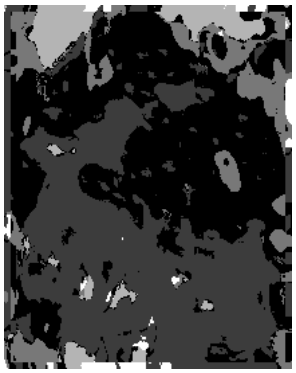
(b) Légende



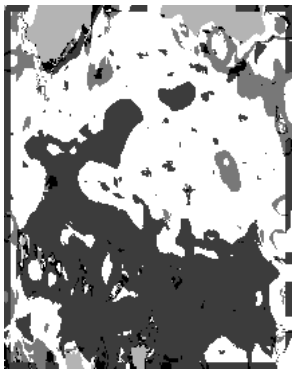
(c) Ensemble d'apprentissage non traité, approche par classes



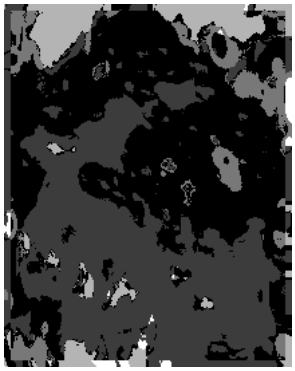
(d) Ensemble d'apprentissage non traité, approche par paires d'attributs



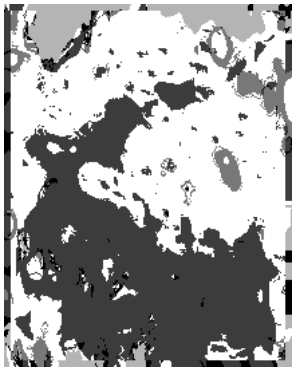
(e) Ensemble d'apprentissage filtré, approche par classes



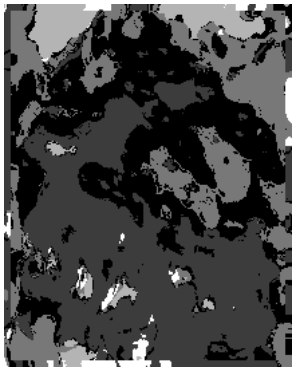
(f) Ensemble d'apprentissage filtré, approche par paires d'attributs



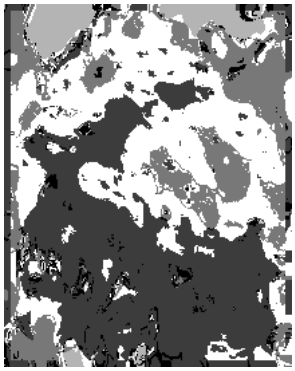
(g) Identification composantes connexes, approche par classes



(h) Identification composantes connexes, approche par paires d'attributs



(i) Les deux traitements, approche par classes



(j) Les deux traitements, approche par paires d'attributs

FIGURE 5.10 – Classification obtenue pour l'application Schneider, section orizontale

aux classes "régions non–orientées" et "régions manque de renfort", il sera comptabilisé dans le cas "ambiguïté négative". Le système permet aussi la situation de "rejet", quand le point n'est classé dans aucune des classes apprises.

Les tableaux 5.1 et 5.2 imposent une remarque générale, valable pour les deux approches : les taux de bonne classification sont très élevés, avec ou sans pré–traitement de l'ensemble d'apprentissage.

Plus particulièrement, l'application de l'algorithme d'identification des composantes connexes a,

pour les deux approches, un effet positif visible : les trois classes sont parfaitement identifiées et individualisées. L'application de l'algorithme d'épuration des points faux a un effet plutôt négatif sur le taux de classification correcte. Par contre, si on se situe dans le contexte réel du problème de classification, ces "erreurs" introduites sont tout à fait explicables. L'ensemble d'apprentissage provient d'une image acquise avec une certaine technologie. Même si cette technologie est très performante, le processus d'acquisition en lui–même introduit certainement du bruit. C'est justement ces valeurs provenant des pixels affectés par le bruit qui sont éliminés de l'ensemble d'apprentissage par l'algorithme d'épuration. Ces pixels, localement isolés, ne sont pas retirés par les experts des ensembles d'évaluation, car ils sont inclus dans un contexte global. Ceci explique le taux de rejet un peu plus élevé pour ce cas.

TABLE 5.1 – Matrices de confusion obtenues pour l'approche par classes

pré-traitement	orienté(%)	non–orienté(%)	manque de renfort(%)	Ambig. positive(%)	Ambig. négative(%)	Rejet(%)
Aucun	100	0.00	0.00	0.00	0.00	0.00
	0.00	100	0.00	0.00	0.00	0.00
	0.00	0.00	99.60	0.40	0.00	0.00
Epuration (1)	99.85	0.00	0.00	0.00	0.00	0.15
	0.00	100	0.00	0.00	0.00	0.00
	0.52	2.17	94.41	2.57	0.07	0.26
Comp. connexes (2)	100	0.00	0.00	0.00	0.00	0.00
	0.00	100	0.00	0.00	0.00	0.00
	0.00	0.00	100	0.00	0.00	0.00
(1) et (2)	99.32	0.00	0.00	0.00	0.00	0.68
	0.00	100	0.00	0.00	0.00	0.00
	0.72	0.79	88.75	7.57	0.79	1.38

L'analyse des matrices de confusion obtenues en appliquant la méthode leave–one–out pour la classe "manque de renfort" mène à la conclusion que cette classe est moins individualisée que les deux autres, mais les taux de classification correcte restent quand même élevés, puisqu'ils sont supérieurs à 85%. L'approche est ensuite évalué en généralisation sur les pointés de test. Les experts ont pointé d'autres régions typiques sur les images 3D disponibles et des vecteurs d'attributs ont été calculés sur les voxels de ces régions. Ces vecteurs ont servi comme entrées dans le système de classification. Les résultats sont présentés dans les tableaux 5.3 et 5.4.

On remarque une très bonne détection de la classe "régions orientées" pour tous les niveaux de pré–traitement, en particulier pour l'approche par paires d'attributs (taux de classification correcte généralement supérieur à 90%), mais également pour l'approche par classes (taux de classification correcte supérieur à 85%). Par contre, pour les deux autres classes la capacité de généralisation du système de classification semble plus faible, surtout pour la classe "non-orientée" qui est pratiquement noyée dans la classe "orientée".

Ces résultats peuvent s'expliquer en regardant la distribution des points d'apprentissage d'une

TABLE 5.2 – Matrices de confusion obtenues pour l'approche par paires d'attributs

pré-traitement	orienté(%)	non–orienté(%)	manque de renfort(%)	Ambig. positive(%)	Ambig. négative(%)	Rejet(%)
Aucun	100	0.00	0.00	0.00	0.00	0.00
	0.00	100	0.00	0.00	0.00	0.00
	0.00	0.00	99.60	0.40	0.00	0.00
Epuration (1)	99.86	0.00	0.00	0.00	0.00	0.14
	0.00	100	0.00	0.00	0.00	0.00
	0.53	2.17	94.41	2.56	0.07	0.26
Comp. connexes (2)	100	0.00	0.00	0.00	0.00	0.00
	0.00	100	0.00	0.00	0.00	0.00
	0.00	0.00	100	0.00	0.00	0.00
(1) et (2)	99.32	0.00	0.00	0.00	0.00	0.68
	0.00	100	0.00	0.00	0.00	0.00
	0.72	0.79	88.75	7.57	0.79	1.38

TABLE 5.3 – Matrices de confusion obtenues pour l'approche par classes, généralisation

pré-traitement	orienté(%)	non–orienté(%)	manque de renfort(%)	Ambig. positive(%)	Ambig. négative(%)	Rejet(%)
Aucun	87.78	0	0.61	10.61	1.00	0.00
	1.44	13.70	0	84.86	0	0
	2.32	16.00	71.70	9.66	0.32	0
Epuration (1)	92.07	0.78	0.30	6.32	0.38	0.15
	1.44	13.89	0.00	84.67	0.00	0.00
	1.14	21.20	69.64	6.63	1.07	0.32
Comp. connexes (2)	88.28	0	0.59	10.12	1.00	0.00
	1.24	17.62	0.00	81.13	0.00	0.00
	2.00	17.14	71.35	8.80	0.61	0.10
(1) et (2)	85.40	4.58	0.45	6.15	0.66	2.76
	1.44	35.25	0.00	63.31	0.00	0.00
	0.32	23.45	64.15	7.31	0.75	4.03

part et des points de l'ensemble de généralisation d'autre part.

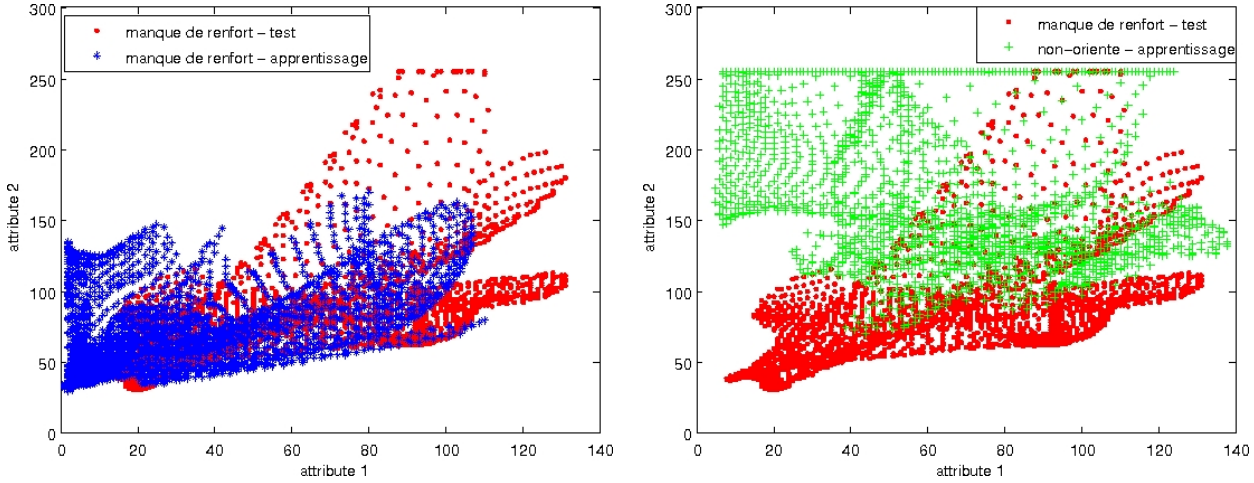
TABLE 5.4 – Matrices de confusion obtenues pour l'approche par paires d'attributs, généralisation

pré-traitement	orienté(%)	non–orienté(%)	manque de renfort(%)	Ambig. positive(%)	Ambig. négative(%)	Rejet(%)
Aucun	99.77	0.00	0.00	0.02	0	0.21
	0.29	12.07	0.00	0.10	0.00	87.55
	2.49	12.08	71.35	0.53	14.00	13.40
Epuration (1)	99.47	0.00	0.00	0.32	0.00	0.21
	1.44	13.89	0.00	0.19	0	84.48
	1.53	17.68	70.13	1.18	0.89	8.59
Comp. connexes (2)	99.74	0.00	0.00	0.04	0.00	0.21
	1.34	15.52	0.00	0.10	0	83.05
	2.39	13.15	71.28	0.57	0.25	12.37
(1) et (2)	92.28	0.28	0.00	7.26	0.00	0.19
	1.44	35.25	0.00	0.19	0	63.12
	0.53	22.06	68.14	2.89	0.85	5.52

Pour la classe de "manque de renfort", on remarque que le taux d'ambiguïté et de rejet est faible, mais que le taux des points affectés à la classe "non–orientée" est très important. En tenant compte du principe de classification implémenté par la méthode proposée, cette situation correspond au cas où une partie des points à classer se situent d'une part en dehors des nuages définis pour la classe "manque de renfort" par l'ensemble d'apprentissage et d'autre part à l'intérieur des nuages définis pour la classe "non–orientée" dans le même ensemble. Une analyse par paires d'attributs de la distribution des points des deux nuages (de test et d'apprentissage) a été réalisée afin de confirmer cette hypothèse.

La figure 5.11 présente une comparaison entre le nuage des points représentant la classe "manque de renfort" de l'ensemble de test superposé d'une part avec le même nuage de l'ensemble d'apprentissage (Fig. 5.11(a)), puis d'autre part avec le nuage représentant la classe non–orientée de l'ensemble d'apprentissage (Fig. 5.11(b)), et ce pour les attributs A_1 et A_2. On peut ainsi remarquer que la superposition des deux nuages (apprentissage et test) pour la classe manque de renfort est loin d'être parfaite, le nuage de test n'étant pas couvert par les points d'apprentissage dans ses extrémités haute et à droite. Les points de l'extrémité haute du nuage de test ($A_1 < 115$, $A_2 > 180$) sont par contre entièrement couverts par le nuage d'apprentissage pour la classe non–orientée. De plus, une partie des points de l'extrémité droite sont également couverts par le nuage d'apprentissage de la classe non–orientée. Naturellement, la décision de classification de ces points pour cette paire d'attributs sera la classe non–orientée.

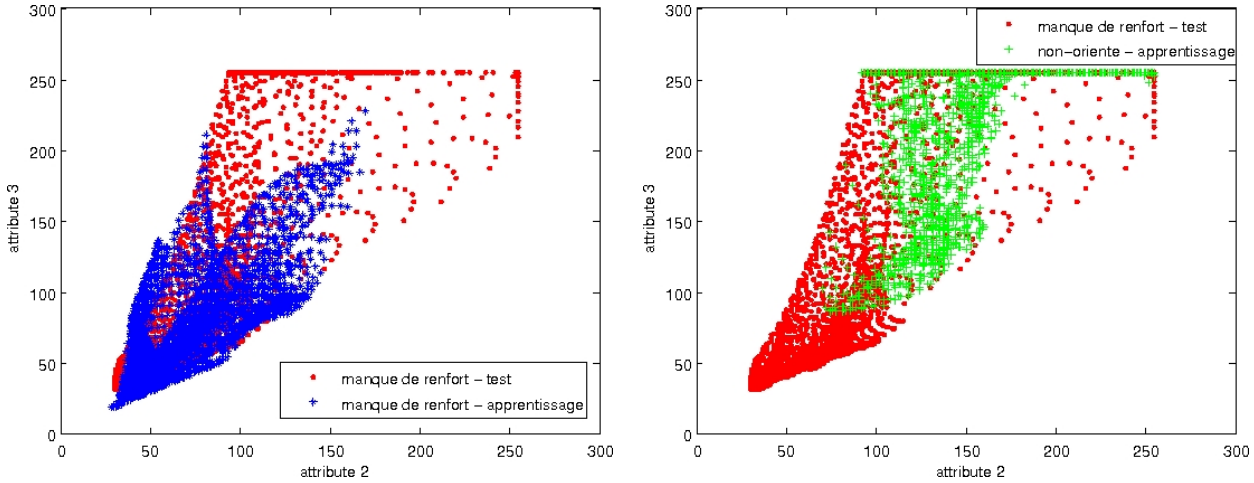
La figure 5.12 présente une comparaison similaire, mais pour les attributs 2 et 3, et elle peut être analysée de la même manière. La classe non–orientée récupère les points situés dans l'extrémité supérieure du nuage de test ($A_3 > 210$, $A_2 \in [70, 170]$) ainsi que tous les points qui proviennent évidemment d'une saturation du processus de création des images des attributs (difficulté liée au calage de la dynamique des attributs lors de leur création). Ces points seront ainsi affectés à la classe



(a) Ensemble de test vs ensemble d'apprentissage – classe manque de renfort, attributs 1 et 2

(b) Classe manque de renfort dans l'ensemble de test vs. classe non–orienté dans l'ensemble d'apprentissage, attributs 1 et 2

FIGURE 5.11 – Comparaison des nuages de l'ensemble d'apprentissage et de l'ensemble de test non–orientée pendant le processus de classification.



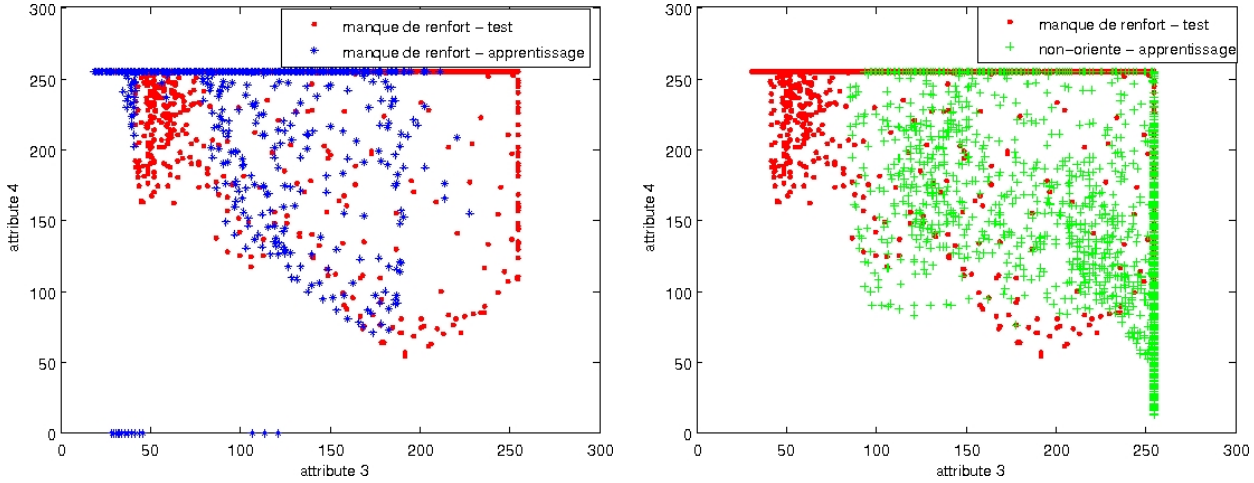
(a) Ensemble de test vs ensemble d'apprentissage – classe manque de renfort, attributs 2 et 3

(b) Classe manque de renfort dans l'ensemble de test vs. classe non–orienté dans l'ensemble d'apprentissage, attributs 2 et 3

FIGURE 5.12 – Comparaison des nuages de l'ensemble d'apprentissage et de l'ensemble de test

Une dernière paire d'attributs analysés est formée des attributs 3 et 4 et la comparaison visuelle est présentée dans la figure 5.13. Le nuage de l'ensemble d'apprentissage pour la classe manque de renfort ne couvre pas la partie droite du nuage de test, alors que le nuage de la classe non–orientée couvre entièrement cette partie ($A_3 > 230$, $A_4 > 80$). Encore une fois, des points situés dans cette région seront affectés, selon les règles de classification, à la classe non–orientée.

Le même type d'analyse peut être réalisée pour les autres paires d'attributs, avec des résultats similaires. En conclusion, la classification erronée d'un pourcentage assez important des points de la classe manque de renfort dans la classe non–orientée est expliquable par la distribution des points d'apprentissage par rapport aux points de test. La différence en distribution est probablement dûe



(a) Ensemble de test vs ensemble d'apprentissage – classe manque de renfort, attributs 3 et 4

(b) Classe manque de renfort dans l'ensemble de test vs. classe non–orienté dans l'ensemble d'apprentissage, attributs 3 et 4

FIGURE 5.13 – Comparaison des nuages de l'ensemble d'apprentissage et de l'ensemble de test

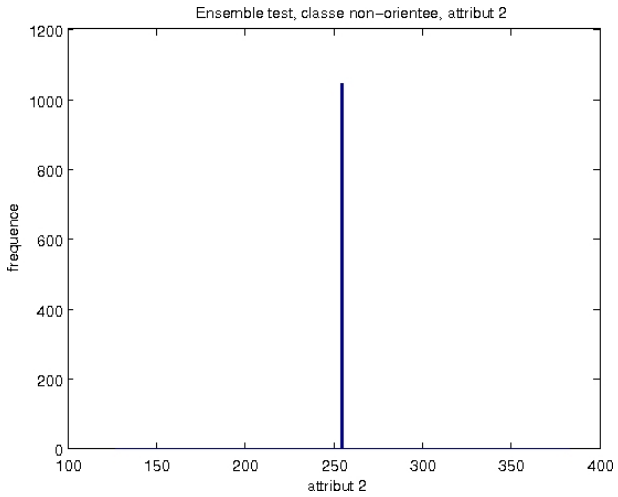
aux causes qui forment les régions recherchées. Le même type de région (manque de renfort par exemple) peut provenir d'une multitude de phénomènes mécaniques et/ou chimiques et donc même si d'un point de vue structurel elles sont similaires, elles peuvent répondre différemment au processus d'extraction des attributs.

De même, un second phénomène nécessite d'être expliqué. Il s'agit de la très mauvaise détection de la classe "non–orientée". La grande majorité des points de cette classe est placée dans l'ambiguïté, notamment entre la classee non–orientée et la classe orientée. L'analyse des histogrammes des points de test de cette classe (voir fig. 5.14(a) et 5.15(a)) a montré une saturation complète des attributs 2 et 3.

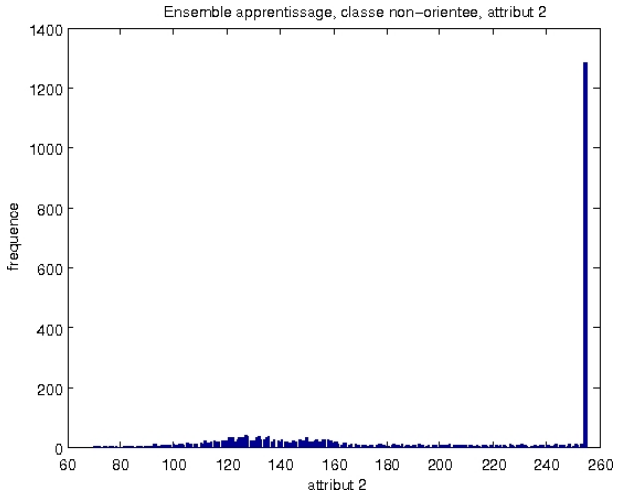
Comme on peut remarquer d'une part sur les figures 5.14(b) et 5.14(c) et d'autre part sur les figures 5.15(b) et 5.15(c), les ensembles d'apprentissage des classes orientée et non–orientée incluent cet unique point $(A_2, A_3) = (255, 255)$ qui constitue la totalité de l'ensemble de test pour ces 2 attributs. On peut donc conclure qu'une partie des "boîtes élémentaires" qui constituent le système d'apprentissage va certainement voter pour le classement des points de test dans l'ambiguïté.

Pourtant, les ensembles d'apprentissage des deux classes sont bien identifiées et individualisées par le système de classification, comme le prouvent les résultats présentés dans les tableaux 5.1 et 5.2. La figure 5.16(b) illustre le fait que la séparation entre les deux classes est donnée principalement par l'attribut 1. Cet attribut a des valeurs plutôt élevées pour la classe non–orientée et des valeurs plutôt faibles pour la classe orientée. Pourtant, il existe une superposition entre les 2 nuages, notamment pour l'attribut 1 compris entre ~ 20 et ~ 70. Or en analysant la distribution des points de test de la classe non–orientée pour l'attribut 1 (Fig. 5.16(a)), on se rend compte que $\sim 70\%$ des points ont la valeur de l'attribut 1 comprise justement dans cet intervalle. En tenant compte des distributions des attributs 2 et 3, qui ont été déjà analysées, on peut conclure que, étant donné l'ensemble d'apprentissage, le système de classification peut au mieux séparer le reste de 30% des points de l'ensemble de test, ce qui est d'ailleurs le cas pour les deux approches quand les deux types de pré–traitement des données d'apprentissage sont appliqués.

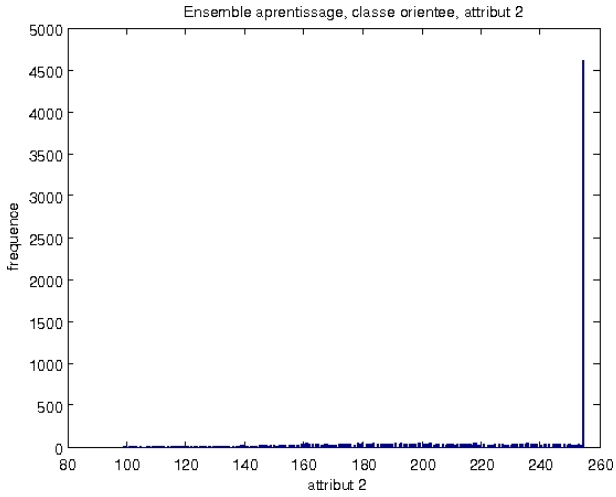
L'analyse des images tomographiques est une application complexe, car il est difficile de définir précisement les régions recherchées et de plus il n'existe pas vraiment de mesure objective absolue qui



(a) Histogramme ensemble de test, classe non–orientée



(b) Histogramme ensemble d'apprentissage, classe non–orientée

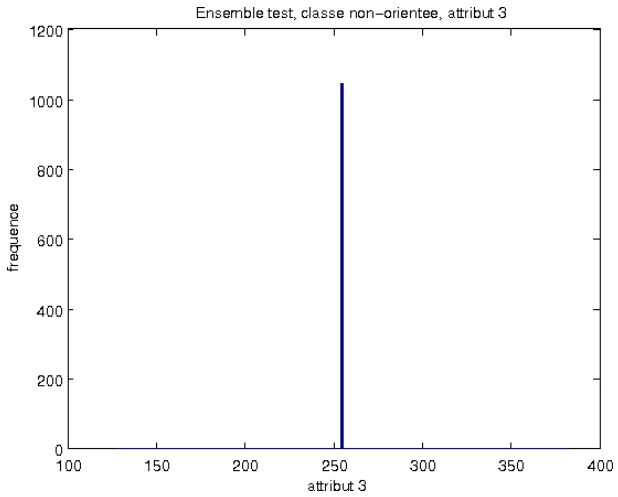


(c) Histogramme ensemble d'apprentissage, classe orientée

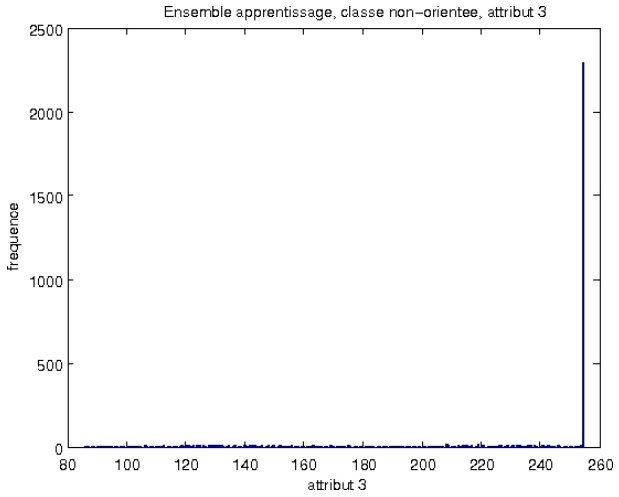
FIGURE 5.14 – Comparaison des histogrammes pour l'attribut 2

dicte dans quelle région se situe un voxel donné. L'application du sytème de classification proposé sur cette application offre de l'information pertinente aux experts, en associant une étiquette à chaque voxel des images fournies. Même les étiquettes indiquant l'ambiguïté apportent de l'information dans le système et donnent une image d'ensemble de la répartition des fibres qui composent les pièces.

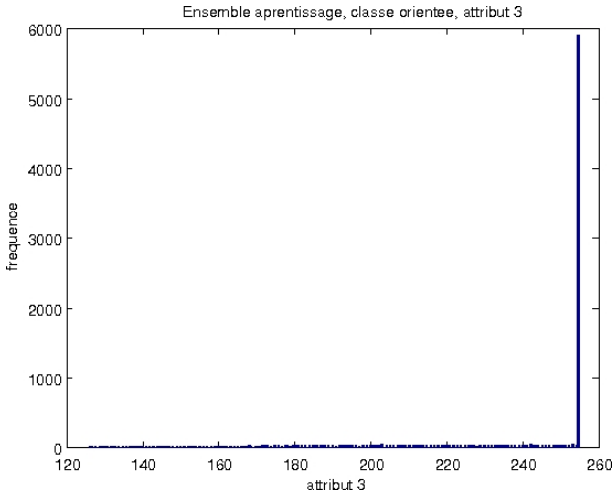
En conclusion, le système de classification développé est un système avec un grand pouvoir de représentation des données d'apprentissage, mais avec un pouvoir de généralisation moins bon quand les données d'apprentissage et les données de test présentent des incohérences. Comme tout système automatique et/ou informatique qui manque d'intelligence intrinsèque, il est capable d'apprendre seulement selon le modèle fourni pour l'apprentissage, et il ne peut pas étendre les notions qu'il a appris dans un contexte plus général. En général, on peut conclure qu'une différence importante entre les taux de classification correcte sur l'ensemble d'apprentissage et sur l'ensemble de test correspond à une incohérence entre les deux et indique la nécessité d'une analyse plus aprofondie de ces ensembles. Comme cette incohérence est souvent inévitable, des travaux futurs visant à rajouter de la gradualité dans la sortie autour de la zone délimitée par l'ensemble d'apprentissage peuvent être envisagés afin de placer les points de ces régions dans la classe analysée, mais avec un degré d'appartenance inférieur à 1.



(a) Histogramme ensemble de test, classe non–orientée



(b) Histogramme ensemble d'apprentissage, classe non–orientée



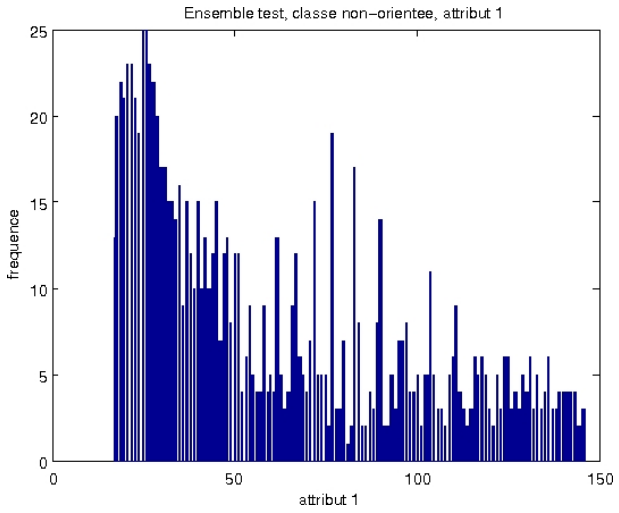
(c) Histogramme ensemble d'apprentissage, classe orientée

FIGURE 5.15 – Comparaison des histogrammes pour l'attribut 3

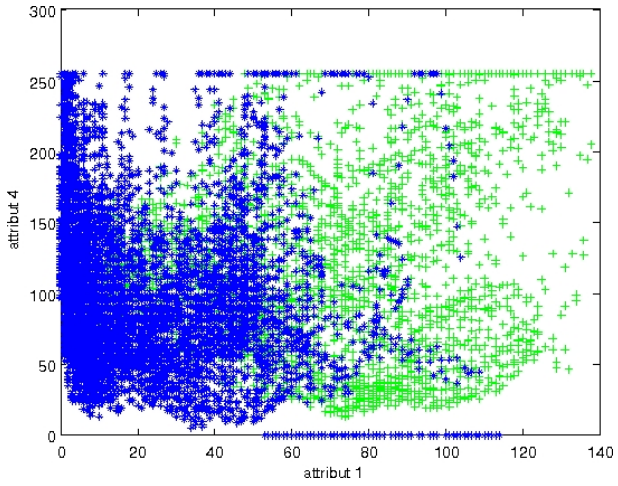
5.2 Imagerie radar

Les images satellitaires radar sont très utilisées dans diverses applications, notamment pour étudier des zones géographiques difficiles d'accès. Les images radar peuvent être obtenues en utilisant plusieurs types de radars. Parmi les choix possibles, les radars SAR (Synthetic Aperture Radar) sont particulièrement adaptés à des conditions météorologiques difficiles et ils permettent également l'inspection du sous–sol. L'application des règles graduelles dans le domaine des images satellitaires porte sur la classification des images qui ont été obtenues avec ce type de radar, notamment des acquisitions SAR interférométriques et polarimétriques. Ces types d'acquisitions résultent d'images complexes, multi-canaux, qui de plus nécessitent un processus de traitement assez important. Les traitements nécessaires se décomposent en deux grandes étapes :

– L'extraction de l'information : obtenir de l'information pertinente à partir de l'acquisition directe des images. Cette étape peut être réalisée en utilisant des méthodes spécifiques disponibles dans des logiciels commercialisés [123]



(a) Histogramme ensemble de test, classe non–orientée, attribut 1



(b) Nuages des points d'apprentissage, classe orientée vs. classe non–orientée, attributs 1 et 4

FIGURE 5.16 – Illustration de la superposition des classes orientée et non–orientée

– La fusion de l'information : en fonction de l'application, différentes approches peuvent être envisagées. Une de ces approches est d'obtenir la classification des régions de ces images en utilisant des règles graduelles

Dans [89], les auteurs proposent une comparative de cinq méthodes statistiques de classification non-supervisée appliquées dans le domaine de l'imagerie POLSAR.

L'approche proposée est complémentaire à ces méthodes. Elle a une base essentiellement supervisée, donc des exemples d'apprentissage seront nécessaires afin de démarrer le processus. Par contre, la classification ainsi obtenue aura besoin de beaucoup moins de vérifications supplémentaires et de validations successives. De plus, l'ensemble des règles obtenues peut être utilisé sur n'importe quelle autre image satellitaire qui a été acquise dans les mêmes conditions et avec les mêmes paramètres d'acquisition.

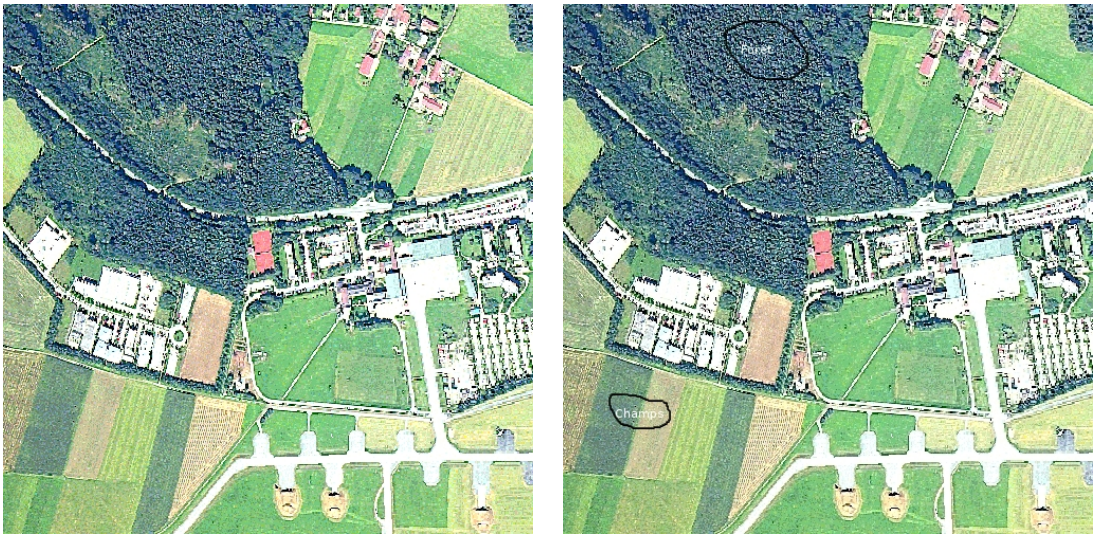
5.2.1 Aéroport Oberpfaffenhofen

5.2.2 Problématique

Les données radar de l'aéroport Oberpfaffenhofen ont été utilisées dans différentes applications de classification dans le domaine de l'imagerie satellitaire [130, 57]. Le système de classification proposé a été également appliqué sur les données Oberpfaffenhofen uniquement afin d'établir l'applicabilité du principe proposé aux données radar.

L'image optique correspondant à la région analysée par le système est présentée sur la figure 5.17(a).

Les données disponibles sont des images en niveaux de gris sur 8 bits, représentant différents attributs spécifiques couramment utilisés dans le domaine : l'entropie H, l'angle de réfraction de l'onde radar α et l'amplitude [133]. Le système éveloppé dans cette thèse a été appliqué sur ces données afin de réaliser une classification des différentes régions, notamment l'identification des zones couvertes par des forêts et des zones agricoles (champs couverts par différents types de cultures).

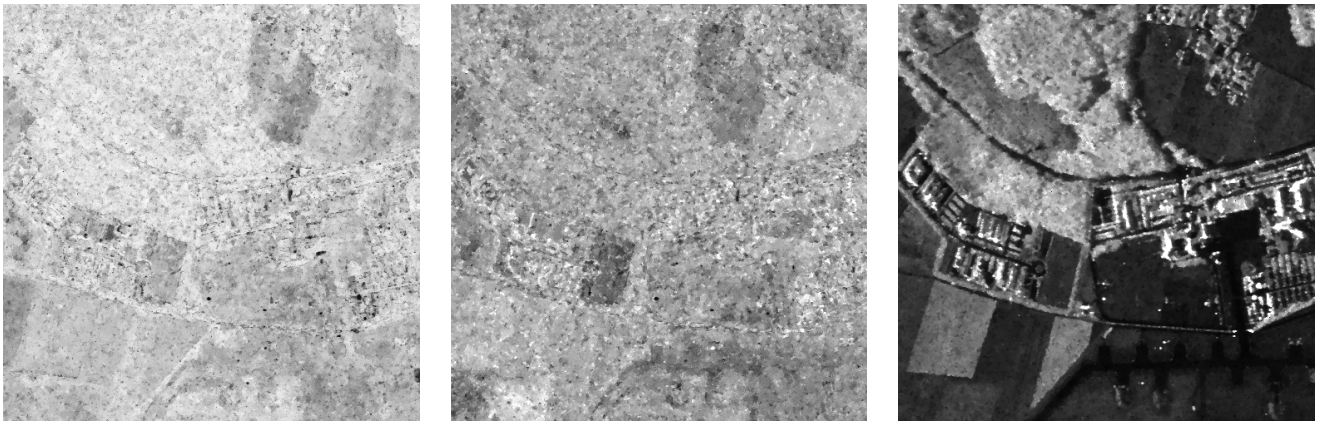


(a) Image optique (b) Régions typiques pointées par les experts

FIGURE 5.17 – Aéroport Oberpfaffenhofen

Les règles de classification ont été déduites à partir des régions typiques pointées par les experts sur l'image optique, comme montré dans l'image 5.17(b). Ces zones ont été identifiées dans les images correspondant aux trois attributs utilisés et les valeurs obtenues ont servi comme ensemble d'apprentissage.

Les images des trois attributs utilisés sont présentées dans la figure 5.18, où 5.18(a) représente l'entropie, la figure 5.18(b) l'angle de réfraction et la figure 5.18(c) l'amplitude [72, 70, 93, 18, 49].



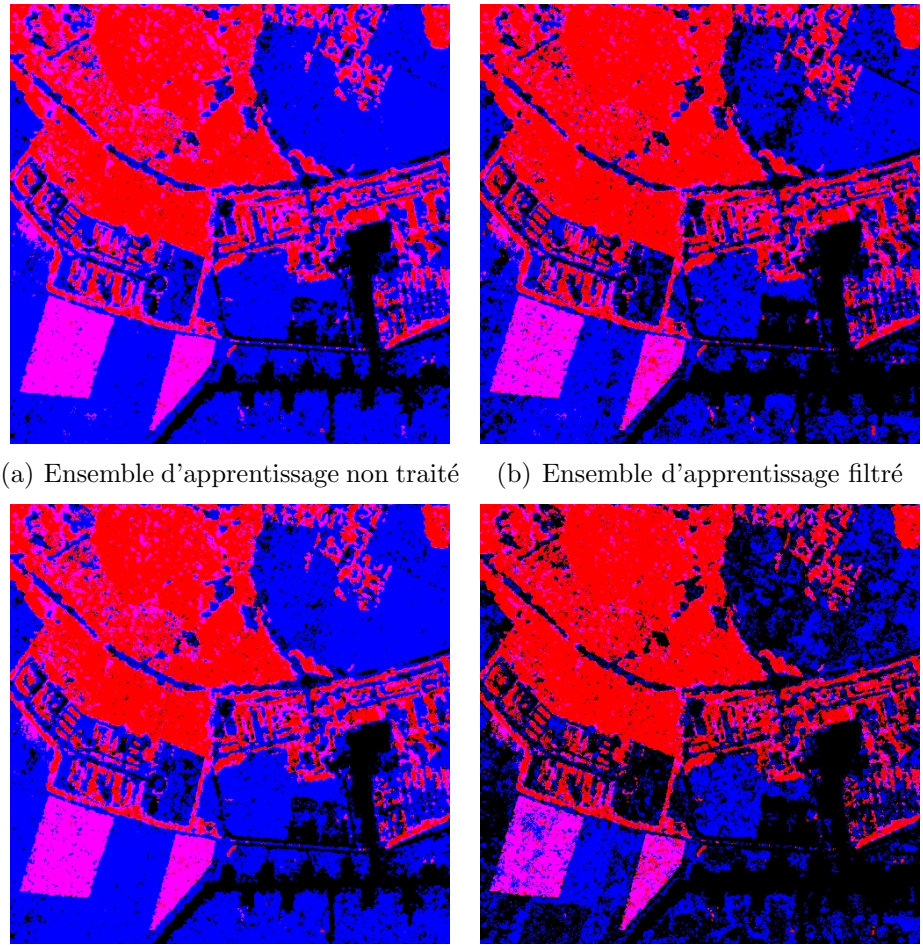
(a) Entropie H (b) Angle α (c) Amplitude A

FIGURE 5.18 – Attributs utilisés

Le paragraphe suivant présente les résultats qualitatifs (images représentant les résultats globaux avec les classes codées en couleurs) et quantitatifs (matrices de confusion obtenues sur les ensembles d'apprentisssage en utilisant la méthode d'évaluation "leave–one–out") pour les deux types d'exploitation des règles : approche par classe et approche par paire d'attributs.

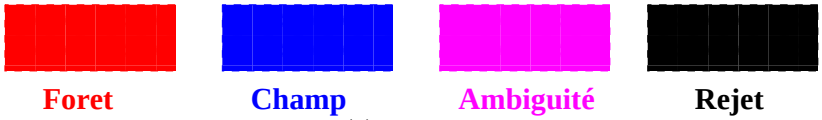
5.2.2.1 Résultats pour l'approche par classes

La figure 5.19 présente le résultat de classification en appliquant le système de classification développé pour l'approche par classes. D'une manière générale, on remarque une bonne identification des zones recherchées pour les quatre niveaux de pré–traitement, ainsi qu'un bon rejet des zones qui ne correspondent pas aux classes apprises, comme la route avec les parkings en bas à droite de l'image.



(a) Ensemble d'apprentissage non traité (b) Ensemble d'apprentissage filtré

(c) Identification composantes connexes (d) Les deux traitements



(e) Légende

FIGURE 5.19 – Classification – approche par classes

L'influence de l'épuration de l'ensemble d'apprentissage mène à l'augmentation du nombre de pixels non–classifiés, mais aussi à la diminution du nombre de pixels classifiés en ambiguïté (voir les deux champs en bas à gauche). Même si l'application de l'algorithme d'identification des composantes connexes n'apporte pas de résultats significatifs quand il est appliqué seul, on peut remarquer l'augmentation significative des pixels classifiés dans le rejet quand les deux types de pré-traitement sont enchaînés. Le résultat obtenu dans cette situation est moins bon que pour le cas de l'ensemble de données non–traité, notamment pour l'identification des champs en haut à droite. Pourtant, ce résultat est expliquable par la distribution des valeurs des attributs dans la zone de référence corres-

pondant aux champs. Cette distribution est montrée dans la figure 5.20 où deux pics sont clairement identifiables pour l'attribut A. Si on se rapporte à la figure 5.17(b), il apparait que les deux types de champs qui ont été choisis pour servir d'ensemble d'apprentissage ont une réaction très différente aux ondes radar pour cette composante.

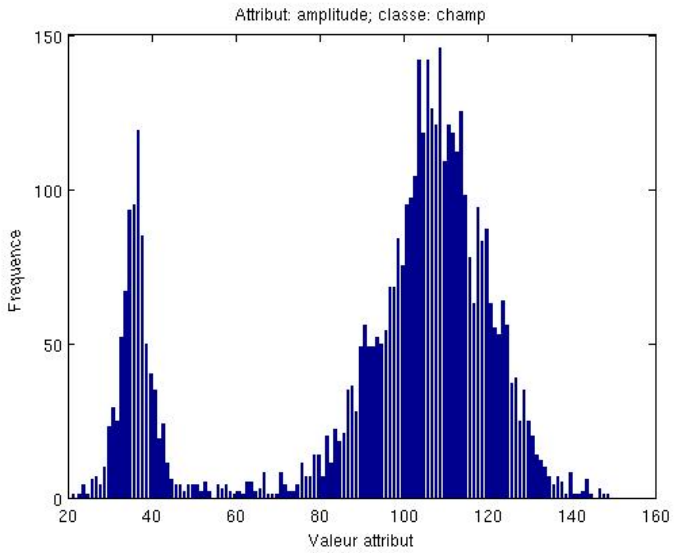


FIGURE 5.20 – L'histogramme des points d'apprentissage pour la classe "champ"

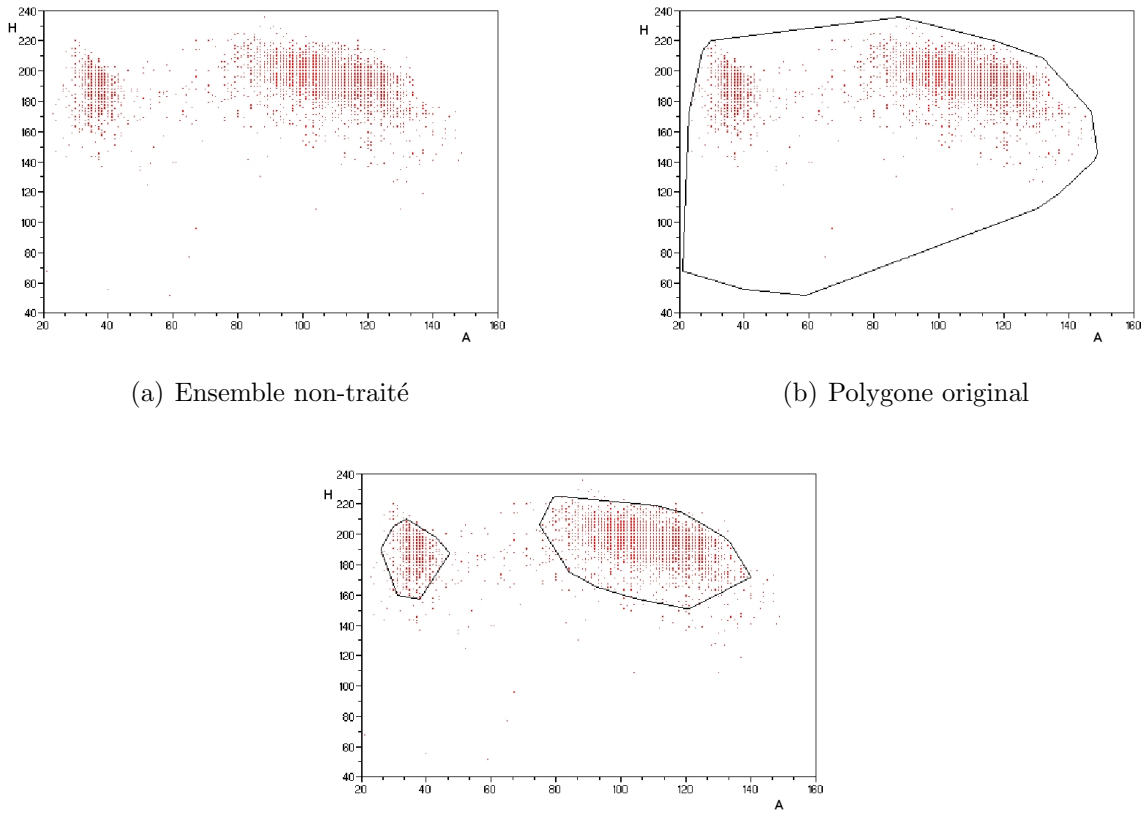
La figure 5.21(a) présente l'ensemble d'apprentissage pour la classe "champ" dans l'espace des attributs entropie (H) et amplitude. Les points s'organisent dans des formes similaires pour les attributs alpha et amplitude. La distribution en deux composantes connexes principales est encore une fois bien visible. L'apprentissage des règles sur l'ensemble initial mène à définir dans l'espace des attributs donné le polygone présenté dans la figure 5.21(b). Ce polygone présente deux désavantages majeurs :

- Il englobe des zones dans l'espace des attributs où il n'y a pratiquement pas de points d'apprentissage, puisque les sommets du polygone correspondent à des points qui sont certainement du bruit et qui se situent très loin de la plupart des autres points d'apprentissage.
- Il ne tient pas compte de la structure bimodale de l'ensemble d'apprentissage.

Les deux composantes connexes ne sont pas identifiées par l'algorithme d'identification des composantes connexes parce que les points "faux" qui se situent entre les deux "nuages" rendent leur distinction assez difficile (un réglage très fin des paramètres est nécessaire pour les séparer). Par contre, l'application successive des deux étapes de pré–traitement a comme effet l'obtention des deux polygones représentés dans la figure 5.21(c) : l'épuration des points "faux" élimine d'abord tous les points situés en dehors des polygones de la figure et ensuite l'identification des composantes connexes sépare l'ensemble d'apprentissage en deux sous–ensembles distincts.

On peut donc conclure que même si d'un point de vue lié strictement à l'application et aux résultats attendus l'introduction des différents niveaux de traitement diminue la qualité de la classification, d'un point de vue opérationnel, étant donné les ensembles d'apprentissage, la qualité et le degré de confiance de la classification augmente.

Un autre problème qui peut être remarqué est l'existence d'importantes zones d'ambiguïté, surtout sur les trois types de champs présents en bas de l'image. La figure 5.22 présente les points d'apprentissage des deux classes pour chaque paire d'attributs utilisée, en bleu la classe "champ" et



(a) Ensemble non-traité

(b) Polygone original

(c) Polygones pour ensemble traité

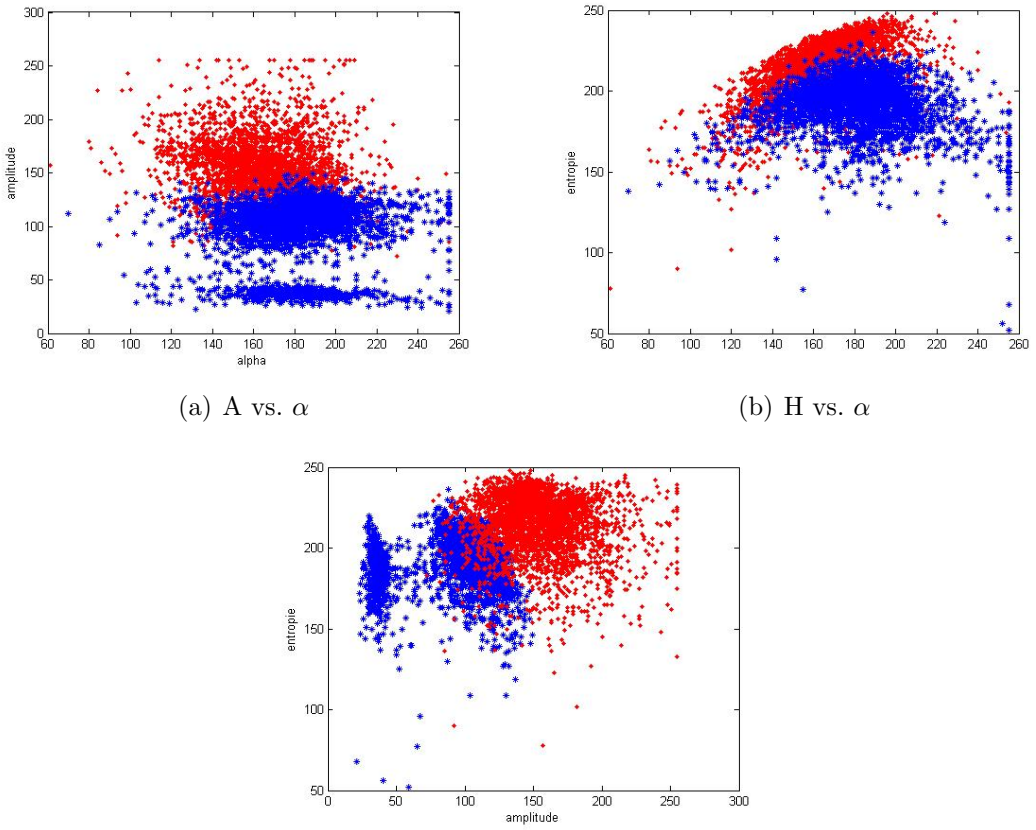
FIGURE 5.21 – Ensemble d'apprentissage pour la classe "champ" (attributs : H/A)

en rouge la classe "forêt". Les nuages correspondant aux deux classes ont une petite partie qui se superpose pour chaque paire d'attributs. Ces zones de superposition correspondent respectivement à des régions de forêt et de champ qui ont des réactions similaires aux ondes radar. Les points de l'image analysée qui correspondent à des attributs qui se situent dans ces régions de chevauchement seront nécessairement placés dans l'ambiguïté entre les deux classes. On peut remarquer que pour la paire d'attributs (α, H) l'ensemble d'apprentissage de la classe "forêt" englobe presque totalement celui de la classe "champ".

Le tableau 5.5 présente les matrices de confusion obtenues pour l'approche par classe pour les quatre niveaux de pré–traitement. Les résultats numériques illustrent très bien les résultats visuels de la figure 5.19. On remarque la bonne identification de la classe "forêt" et le placement massif des points d'apprentissage de la classe "champ" en ambiguïté. La tendance d'amélioration induite par la chaîne de pré–traitements est plus visible grâce à ces matrices, surtout pour la classe problématique des zones de forêt, pour laquelle le taux de classification correcte augmente de 16% avec l'application des deux types de pré–traitement.

5.2.2.2 Résultats pour l'approche par paires d'attributs

La figure 5.23 présente les résultats obtenus en utilisant les même règles de classification par l'approche par paires d'attributs. Les remarques antérieures sont également valables pour ces résultats, sauf que généralement l'ambiguïté de l'approche précédente se transforme en rejet. Ce résultat est



(a) A vs. α

(b) H vs. α

(c) H vs. A

FIGURE 5.22 – Superposition des ensembles d'apprentissage pour les différents paires d'attributs

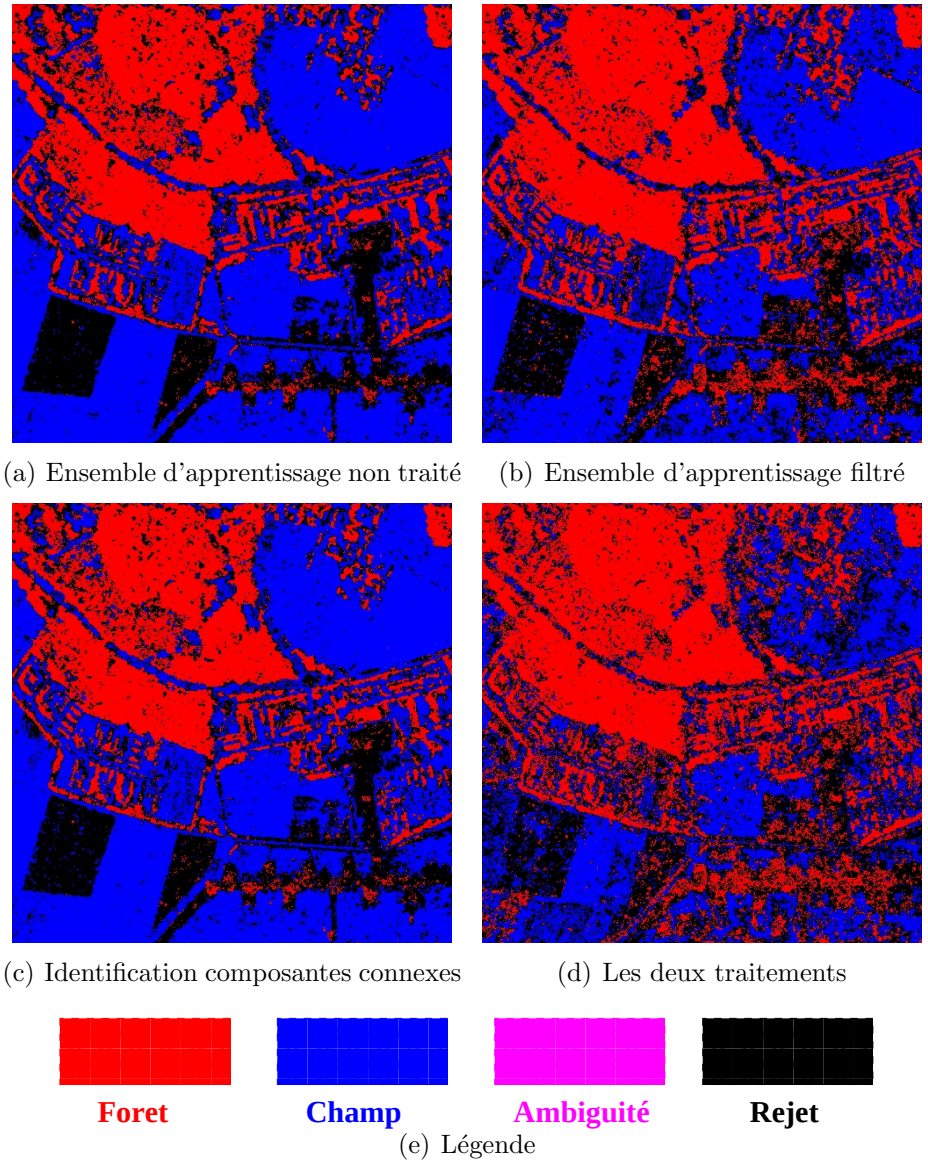
TABLE 5.5 – Matrices de confusion obtenues pour l'approche par classes

Niveau pré-traitement	Forêt (%)	Champ (%)	Ambiguïté (%)	Rejet (%)
-	59.74	0	40.26	0
	0	0.66	99.34	0
Epuration (1)	86.79	0.14	13.06	0
	5.86	4.67	89.19	0.29
Comp. connexes (2)	61.14	0	38.86	0
	0	1.34	98.66	0
(1) et (2)	86.88	0.39	12.73	0
	6.06	16.82	76.41	0.7

prévisible, étant donné le nombre faible (trois) des paires d'attributs utilisées. Cette approche apporte donc une quantité d'information plus faible, mais le résultat peut être considéré comme plus fiable.

Le tableau 5.6 confirme lui aussi les résultats visuels présentés dans la figure 5.23. La comparaison des valeurs des tableaux 5.5 et 5.6 montre une transposition presque parfaite entre les points placés dans l'ambiguïté par la première approche et dans le rejet par la deuxième.

Ces premiers résultats ont montré l'applicabilité de la méthode proposée sur des données issues de



(a) Ensemble d'apprentissage non traité (b) Ensemble d'apprentissage filtré

(c) Identification composantes connexes (d) Les deux traitements

(e) Légende

FIGURE 5.23 – Classification - approche par paires d'attibuts

TABLE 5.6 – Matrices de confusion obtenues pour l'approche par paires d'attributs

Niveau pré-traitement	Forêt (%)	Champ (%)	Ambiguïté (%)	Rejet (%)
-	59.74 0	0 0.66	0 0	40.26 99.34
Epuration (1)	86.79 5.86	0.14 4.67	0 0.59	13.06 88.79
Comp. connexes (2)	61.14 0	0 1.34	0 0	38.86 98.66
(1) et (2)	86.88 6.17	0.39 16.93	0.06 1.43	12.67 75.47

l'imagerie satellitaire radar. La section suivante présente une application radar axée sur l'identification

des bandes de Forbes sur le glacier de Tacul.

5.2.3 Analyse du glacier du Tacul

5.2.3.1 Problématique

Une collaboration internationalle entre le Centre Aérospatial Allemand (DLR) et quatre laboratoires français a été mise en place afin d'acquérir des données POLSAR de haute résolution sur des glaciers situés dans la région Chamonix – Mont Blanc (France). Les données utilisées dans cette thèse ont été obtenues en Octobre 2006 et Février 2007 à l'aide d'un système d'acquisition expérimental du DLR (Experimental Synthetic Aperture Radar System E–SAR). L'ensemble de données utilisé représente une acquisition polarimétrique monostatique complète en bande L sur le glacier du Tacul, un des trois glaciers qui composent le deuxième plus grand complexe glacier d'Europe (la Mer de Glace) [107].

Une des particularités du glacier du Tacul est la présence de bandes de Forbes [48]. Ce phénomène se manifeste par l'alternance des régions blanches et foncées, comme montré dans la figure 5.24.

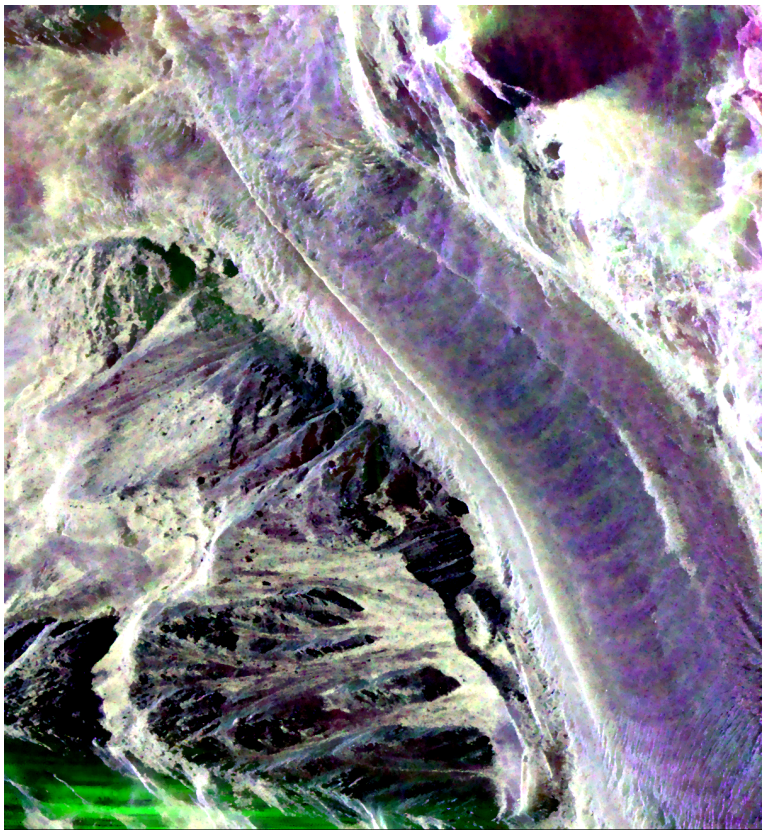
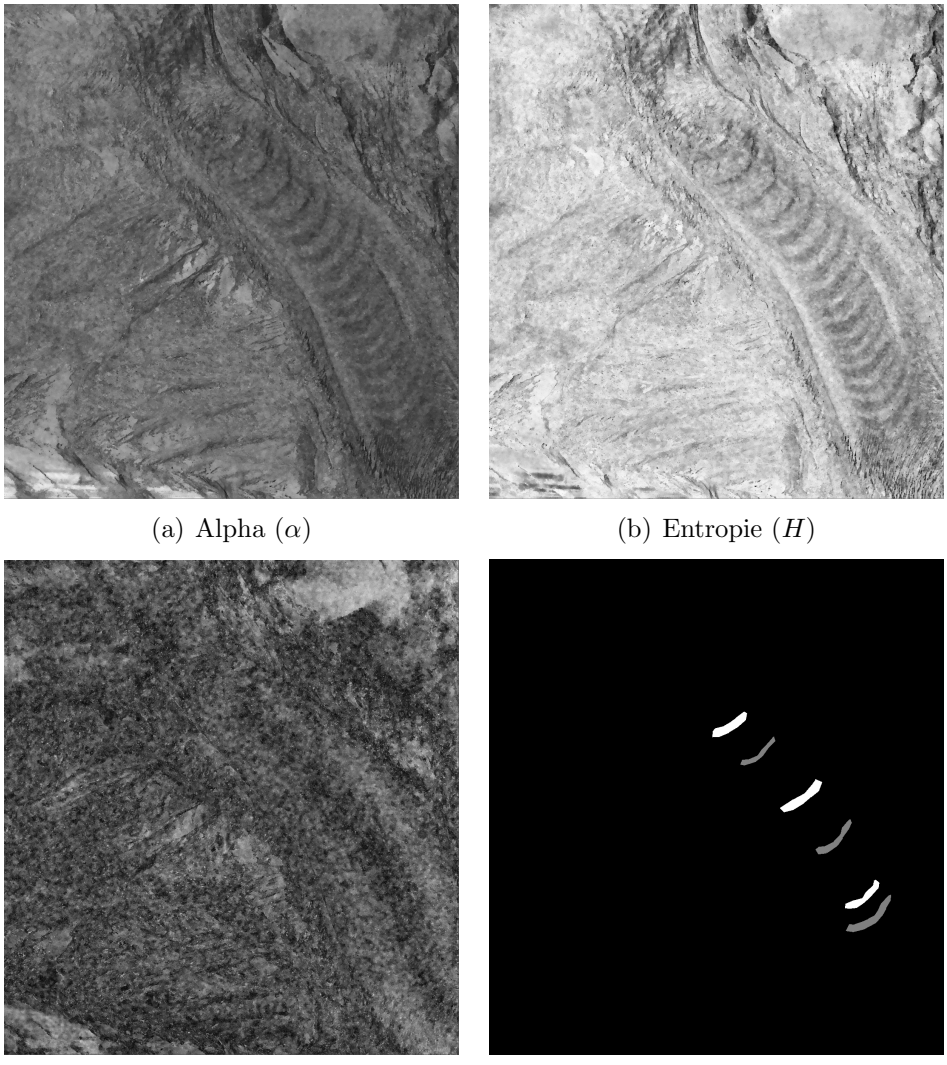


FIGURE 5.24 – Glacier Tacul - présence du phénomène des bandes Forbes

La cause de ce phénomène est encore à déterminer, mais une possibilité généralement acceptée est le fait que la glace glisse plus pendant l'été que pendant l'hiver. La superficie de la glace située au–dessous de la cascade de glace du Géant, située en haut du glacier Tacul, forme une série de terrasses. La neige des terrasses qui sont orientées nord fond moins que celle des terrasses orientées sud. En conséquence, la glace des terrasses orientées nord est plus pure [153]. Cette alternance de glace propre et sale est dûe à la variation de la quantité de poussière minérale (cryoconites) [137]. L'origine de

cette poussière est liée à l'influence de la matière minérale sur la diminution de la température et de la pression de la fusion glacière et donc sur la vitesse de fusion [137, 60]. Les grains de poussière qui sont incorporés dans la glace sont entourés par un film d'eau en état liquide. Ce film induit une réduction de la température locale de fusion de presque 1°C. Ainsi, il a été prouvé dans [60] que la glace impure fond plus vite que la glace propre. En haut du glacier, les bandes de Forbes gardent une direction perpendiculaire sur le glacier, mais en bas leur centre descend plus vite que les côtés. En conséquence, leur forme change, devenant courbe. Cette forme montre les variations de la vitesse de l'avancement de la glace dans les différentes étapes de son parcours, et se révèle donc être un bon indicateur du flux de la vitesse [153].

Le système de classification a été appliqué afin d'obtenir une classification des bandes de Forbes. La figure 5.25 présente les attributs POLSAR qui ont été utilisés : l'entropie H, l'angle α et l'anisotropie. Les ensembles d'apprentissage proviennent des zones de référence pointées par les experts dans les deux classes considérées (neige impure et neige propre), montrées dans la figure 5.25(d).



(a) Alpha (α) (b) Entropie (H)

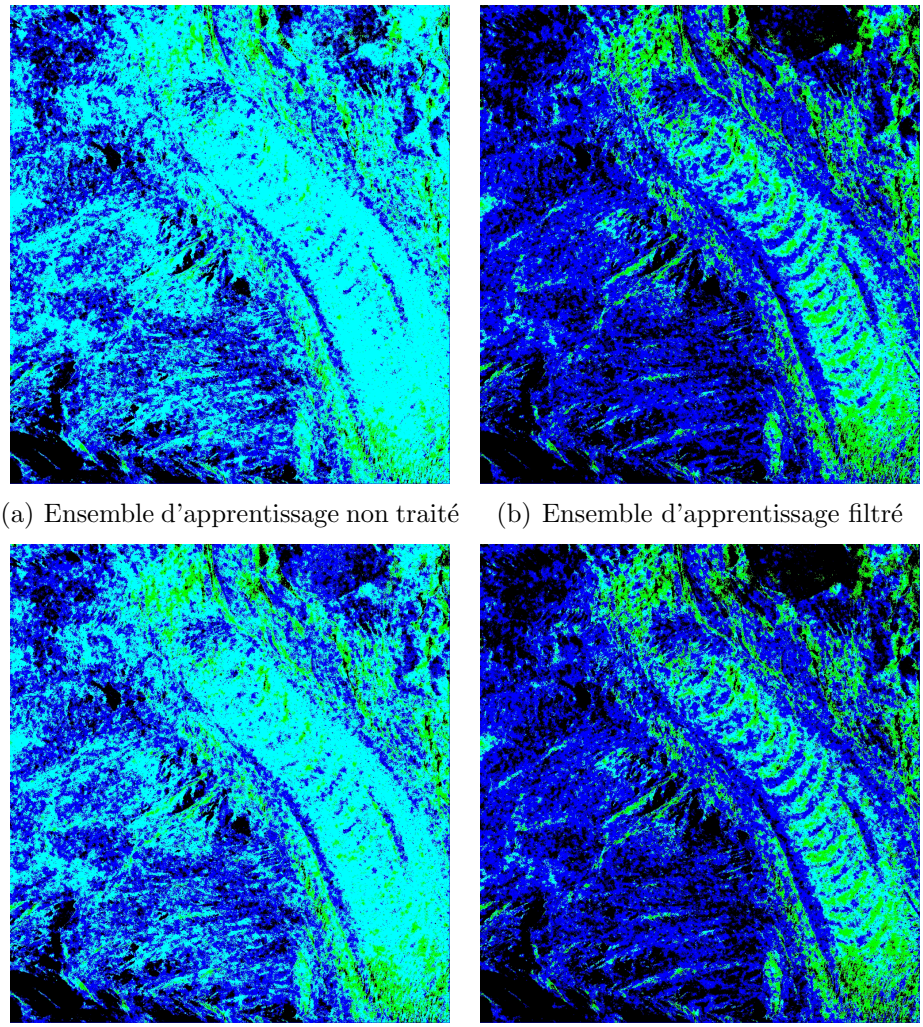
(c) Anisotropie (d) Régions pointées

FIGURE 5.25 – Attributs utilisés pour l'application "Glacier du Tacul"

La section suivante présente les résultats qualitatifs et quantitatifs obtenus en appliquant le système sur les trois attributs pour les deux approches (par classes et par paires d'attributs) et les quatre niveaux de pré–traitement proposés.

5.2.3.2 Résultats

La figure 5.26 présente la classification obtenue à partir des attributs des figures 5.25 pour les quatre niveaux de pré–traitement. A noter l'influence très importante des algorithmes de pré–traitement, particulièrement celui d'épuration des points d'apprentissage "faux".



(a) Ensemble d'apprentissage non traité (b) Ensemble d'apprentissage filtré

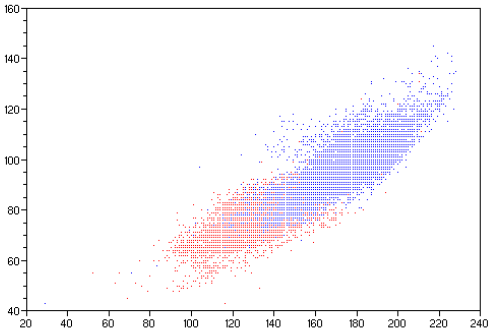
(c) Identification composantes connexes (d) Les deux traitements

(e) Légende

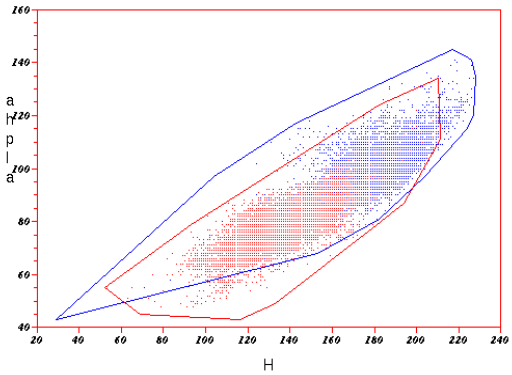
FIGURE 5.26 – Classification - approche par classes

En apprenant les règles de classification sur la totalité de l'ensemble d'apprentissage indiqué par les experts, la quasi–totalité des pixels du glacier sont placés dans une classe d'ambiguïté, le système de classification étant donc incapable de séparer les deux classes. Le phénomène est explicable en regardant la distribution des points d'apprentissage des deux classes, montrée pour la paire d'attributs (H, α) dans la figure 5.27.

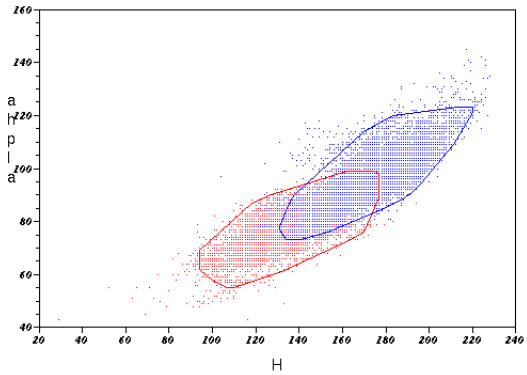
Dans l'image 5.27(a) on peut remarquer que même si le cœur des deux nuages peut être très bien identifié et placé d'une manière intuitive dans une zone compacte, beaucoup de points d'apprentissage sont répandus autour de ces régions compactes où la fréquence des points d'apprentissage est



(a) Nuages d'apprentissage ((α) vs. H)



(b) Polygones englobants originaux



(c) Polygones obtenus pour l'ensemble traité

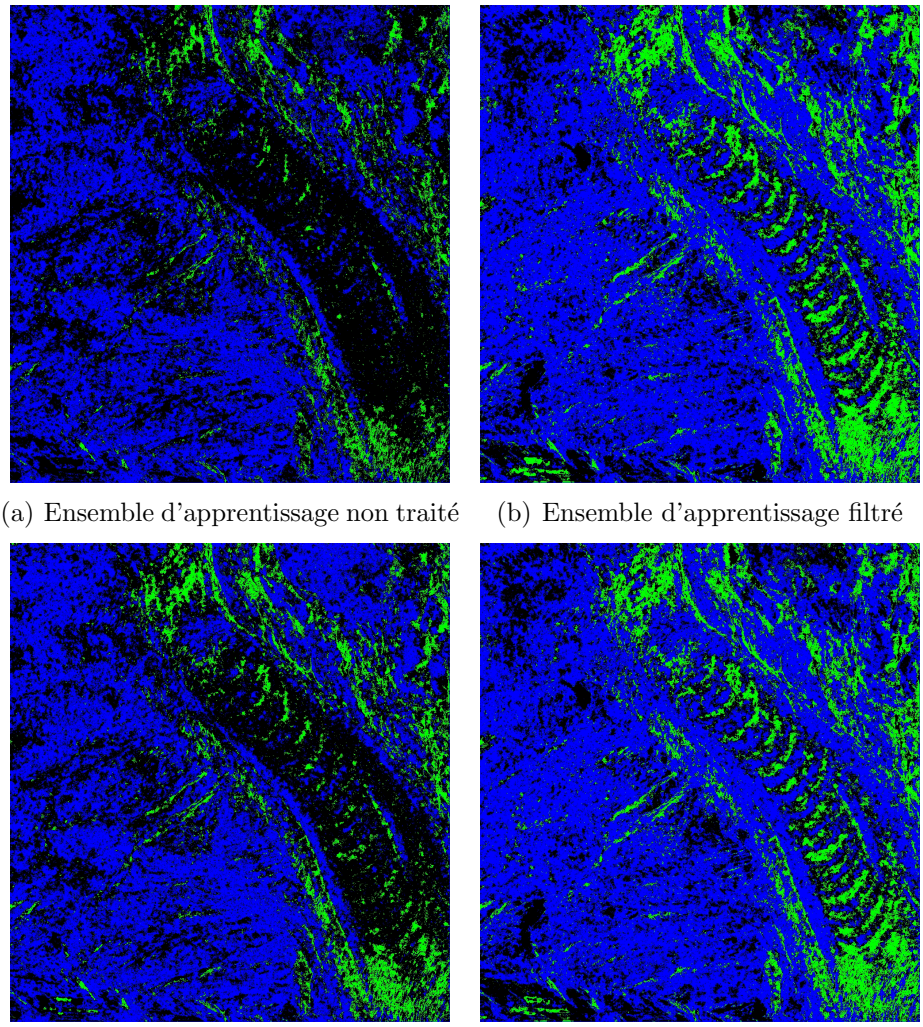
FIGURE 5.27 – Ensembles d'apprentissage de l'application Tacul – attributs H et α

très élevée. La présence de ces points "faux" mène à définir des règles qui décrivent dans l'espace des attributs des polygones englobants qui couvrent une bonne partie de la totalité de l'espace de définition des attributs et dont le degré de superposition est très important, comme montré dans la figure 5.27(b).

L'image 5.27(c) montre les polygones définis par les règles obtenues sur le même ensemble d'apprentissage après l'application de l'algorithme d'épuration des points "faux". Ces polygones sont beaucoup plus petits, ils sont centrés sur la partie des nuages où les points sont très "concentrés", alors que les points qui sont situés loin de ces nuages centraux sont ignorés. Ainsi les nuages couvrent une partie beaucoup plus faible de l'espace des attributs, donc la fiabilité du résultat est augmentée. De plus, en réduisant l'espace assigné à chaque classe, leur superposition est elle aussi beaucoup plus réduite, comme le montre d'ailleurs les résultats présentés dans l'image 5.26(b) par rapport à l'image 5.26(a).

Les petites différences entre les résultats obtenus sur les ensembles d'apprentissage filtrés (figure 5.26(b)) et les ensembles d'apprentissage où les deux algorithmes de pré–traitement ont été appliqués (figure 5.26(d)) sont données par le fait que l'algorithme d'identification des composantes connexes sépare des petites régions avec peu de points d'apprentissage retenus par l'épuration mais qui sont assez loin du corps principal du nuage. Généralement, l'influence de l'application de l'algorithme est assez faible pour cette application, mais elle est positive, car l'ambiguïté est légèrement réduite dans l'image 5.26(c) par rapport à l'image 5.26(a) (et respectivement dans l'image 5.26(d) par rapport à l'image 5.26(b)).

La figure 5.28 représente la classification obtenue pour la même application en appliquant l'approche par paires d'attributs. Même si généralement l'ambiguïté de la première approche devient rejet pour cette approche, diminuant donc la quantité d'informations apportée par le système de classification, on peut remarquer, surtout par la comparaison des images 5.28(d) et 5.26(d), une augmentation du nombre de points correctement placés dans une et une seule classe.



(a) Ensemble d'apprentissage non traité (b) Ensemble d'apprentissage filtré

(c) Identification composantes connexes (d) Les deux traitements

(e) Légende

FIGURE 5.28 – Classification - approche par paires d'attibuts

Ces résultats ont été validés par les experts après une analyse visuelle des images 5.26 et 5.28. Une analyse quantitative a été aussi effectuée. Des matrices de confusion ont été calculées pour les deux approches en utilisant la méthode d'évaluation "leave–one–out". Les résultats sont présentés dans le tableau 5.7 pour l'approche par classes et dans le tableau 5.8 pour l'approche par paires d'attributs. Les valeurs obtenues confirment généralement les résultats visuels des figures 5.26 et 5.28. Pour l'approche par classes, on remarque d'abord le pourcentage énorme de points placés en ambiguïté pour l'ensemble d'apprentissage non–traité, surtout pour la première classe, "neige impure". Le résultat est justifié par la superposition des nuages d'apprentissage, présenté sur la

figure 5.27(b) pour une des trois paires d'attributs. La superposition des nuages est similaire pour les autres deux paires d'attributs. L'épuration des ensembles d'apprentissage augmente significativement les performances du système : le pourcentage des points correctement classifiés dans une et une seule classe augmente de 50% pour la première classe et de 15% pour la deuxième, sans augmentation significative du nombre de points classifiés de manière erronée (rejet ou mauvaise classe). Ce résultat est en accord avec ce qui est attendu du système : les points supprimés par l'épuration sont bien les points situés très loin du cœur du nuage qui provoquaient le degré très important de superposition entre les classes. L'application de l'algorithme d'identification des composantes connexes a beaucoup moins d'influence sur les valeurs de la matrice de confusion.

TABLE 5.7 – Matrices de confusion obtenues pour l'approche par classes – application "Glacier du Tacul"

Niveau pré-traitement	Neige impure (%)	Neige propre (%)	Ambiguïté (%)	Rejet (%)
-	4.75 0.01	0 59.02	95.25 40.8	0 0.17
Epuration (1)	57.07 3.03	1.61 75.6	40.17 19.11	1.15 2.26
Comp. connexes (2)	19.64 0.22	0 54.37	80.36 45.35	0 0.06
(1) et (2)	56.33 2.37	2.15 77.93	39.62 16.88	1.9 2.82

La comparaison des tableaux 5.7 et 5.8 montre que les valeurs obtenues pour l'approche par paires d'attributs correspondent presque parfaitement aux valeurs obtenues pour l'approche par classe, avec la transposition des points placés dans la classe "ambiguïté" par la première approche dans la classe "rejet" par la deuxième. Ce phénomène peut être interpreté comme une diminution de l'information apportée par le système de classification, mais aussi comme une augmentation du degré de confiance associé aux points classés.

TABLE 5.8 – Matrices de confusion obtenues pour l'approche par paires d'attributs – application "Glacier du Tacul"

Niveau pré-traitement	Neige impure (%)	Neige propre (%)	Ambiguïté (%)	Rejet (%)
-	4.75 0.01	0 59.06	0 0.04	95.25 40.88
Epuration (1)	57.95 3.21	1.64 76.34	0.11 1.03	40.29 19.41
Comp. connexes (2)	19.64 0.24	0 54.37	0 0.04	80.36 45.34
(1) et (2)	57.72 2.42	2.29 79.57	0.13 0.27	39.86 17.74

5.3 Conclusions

Ce chapitre montre l'applicabilité des règles floues graduelles comme règles de classification. Deux types d'applications sur lesquelles le système proposé a été appris et ensuite utilisé sont proposés. Les résultats permettent d'identifier de manière systématique et précise des éventuelles incohérences entre les données d'apprentissage et celles de test.

Les résultats sur les deux applications montrent une grande capacité d'apprentissage (évaluée sur l'ensemble d'apprentissage), ainsi qu'une capacité relativement bonne de généralisation sur des données similaires. Les deux méthodes de pré–traitement proposées se montrent très utiles pour les applications où les données sont bruitées (l'épuration) et aussi pour le cas où une classe est caracterisée par une distribution multi–modale dans l'espace des attributs. Malheureusement, le système actuel est très dépendant de l'ensemble d'apprentissage et aucune interpolation n'est possible : de nouvelles données qui ne s'inscrivent pas dans les régions de l'espace des attributs qui sont apprises seront rejetées par le système, même si elles se trouvent dans la proximité immédiate de ces régions. Cet inconvénient pourrait être traité par l'introduction d'une gradualité dans le degré final d'appartenance à la classe, ce qui permettrait d'avoir une représentation floue des contours des polygones convexes.

Conclusions et perspectives

Cette thèse propose un nouveau système de classification, qui se base sur des règles graduelles. Il se situe dans la famille des classifieurs basés sur des règles. Il remplace la forme typique des règles conjonctives "**Si** x_1 *est* A_1 *et* x_2 *est* A_2 **alors** X *appartient à la classe* C *avec le degré* μ" par "**Si** x_1 *est* $A_1 \longrightarrow x_2$ *est* A_2 **alors** X *appartient à la classe* C". L'opérateur d'implication "$\longrightarrow$" est un opérateur particulier qui doit respecter des contraintes de monotonie. L'implication de Rescher–Gaines a été utilisée au cours de ces travaux.

Par rapport aux systèmes de classification basés sur des règles conjonctives, où chaque nouvelle règle apporte de l'information positive, en élargissant l'ensemble des cas qui peuvent être associés à la classe analysée, chaque nouvelle règle implicative apporte de l'information négative, en réduisant l'ensemble des cas qui respectent les contraintes imposées. Les deux types de règles de classification sont donc complémentaires. De cette complémentarité résulte une des contraintes importantes imposées aux systèmes de classification basés sur des règles implicatives : la cohérence. Si une seule règle associée à une classe donnée (ou même une seule contrainte de cette règle) n'est pas cohérente, le résultat sera que la totalité de l'espace des attributs concernés sera placé dans la classe de rejet. Ce problème n'apparaît pas pour les systèmes basés sur des règles conjonctives, où une incohérence a, au pire, comme conséquence un classement erroné d'une partie des points dans la classe analysée. Pour les systèmes implicatifs les éventuelles incohérences sont heureusement détectables a posteriori, en analysant les jeux de règles issues de l'étape d'apprentissage.

Dans la méthode proposée, l'étape d'apprentissage consiste à analyser les données d'apprentissage dans des espaces 2D. Il y a donc autant d'espaces à étudier que de couples d'attributs distincts. L'analyse consiste à identifier les plus petits polygones convexes qui entourent les points d'apprentissage de chaque classe. Ensuite, les règles graduelles qui permettent de délimiter ces polygones sont déterminées pour chaque classe et pour chaque paire d'attributs. Ces règles sont le cœur du système de classification. L'application d'une de ces règles sur un point à classifier donne en fait son appartenance à une classe et pour une paire d'attributs considérés.

L'étape d'utilisation du classifieur ainsi obtenu peut se faire de deux manières. La première approche proposée est l'approche par classes : pour chaque classe, on analyse les sorties des règles correspondant à chaque paire d'attributs et en fonction du nombre de "votes" favorables à la classe (c'est–à–dire de règles de classification validées) on décide de l'appartenance ou non à la classe recherchée. La deuxième approche proposée est l'approche par paires d'attributs. Cette approche fonctionne de la manière suivante : pour chaque paire d'attributs, les règles de classification concernant toutes les classes recherchées, sont appliquées. Si le nombre de règles validées est trop important, c'est–à–dire que le point pourrait potentiellement appartenir à plusieurs classes, alors on considère que l'ambiguïté apportée par le couple d'attributs est trop importante dans la prise de la décision finale. Ainsi seules les couples d'attributs pertinents sont pris en compte. Dans le cas le plus défavorable le système classe le point en rejet.

Parmi les désavantages de la méthode de classification proposée, on peut citer la difficulté d'in-

terprétation des règles exprimées en language naturel, par rapport aux systèmes de classification basés sur des règles conjonctives. On peut également regretter la limitation assez importante liée à l'utilisation même de l'opérateur d'implication, qui se trouve à la base du système. Cet opérateur force l'analyse des attributs disponibles deux par deux. D'un autre côté, une telle analyse a des avantages : l'analyse des nuages des points 2D est toujours plus accessible que l'analyse des nuages multi–dimensionnels et elle permet de traiter les paires d'attributs indépendamment les unes des autres en rejetant les paires qui n'apportent pas de l'information pertinente (approche par paires d'attributs).

Le système ainsi développé, après avoir été validé sur des benchmarks, a été testé sur trois applications réelles :

- **L'interprétation d'images tomographiques 3D :** Le but est d'identifier des régions typiques caractérisant la structure et l'organisation des fibres au sein de pièces électroniques dans des images 3D.
- **Application dans l'imagerie satellitaire, aéroport d'Oberpfaffenhofen :** Cette application très connue dans le domaine de l'analyse des images satellitaires consiste à identifier les champs et les forêts présents dans les images.
- **Application dans l'imagerie satellitaire, l'étude du glacier du Tacul :** Devant les difficultés d'instrumentation des glaciers, l'objectif est de les analyser à distance au moyen des images satellitaires. Les bandes de Forbes du glacier du Tacul ont fait l'objet de cette étude.

Les résultats obtenus sur ces applications ont montré l'intérêt du système développé et la pertinence d'utiliser les règles implicatives comme des règles de classification. Ils ont également nécessité une analyse plus approfondie des données d'apprentissage et/ou de test afin de bien comprendre et interpréter les résultats obtenus.

La complémentarité des deux approches dans la phase d'utilisation du système a également été mise en évidence dans les applications proposées : généralement l'approche par classes est plus tolérante par rapport à l'ambiguïté, alors que l'approche par paires d'attributs est plus stricte de ce point de vue. Exprimée par rapport aux principaux indicateurs calculés sur les matrices de confusion, on peut dire que la première approche est donc plus adaptée aux problèmes où la sensitivité est importante, alors que la deuxième est plus adapté aux problèmes où la précision et la spécificité sont plus importantes.

Suite à la conception et à l'expérimentation de ce système de classification, plusieurs perspectives seraient intéressantes à travailler. La première, déjà évoquée dans le document, porte sur la généralisation de l'opérateur d'implication. Un tel opérateur permettrait d'analyser tous les attributs simultanément et d'obtenir une sortie unique, ce qui faciliterait les traitements liés à la gestion des couples d'attributs. Il est à noter qu'étant donnée la définition même de l'opérateur d'implication, aucune méthode de généralisation n'a été proposée pour le moment.

Des travaux futurs certainement plus accessibles portent sur le rajout de la gradualité dans la sortie du système. Deux approches sont envisageables pour le moment :

- l'utilisation d'un autre opérateur d'implication que l'opérateur de Rescher–Gaines. Différents opérateurs, comme celui de Luksiewicz ou de Willmott offrent des sorties non binaires qui peuvent résulter en une gradualité dans l'appartenance à la classe analysée. Une étude plus avancée est nécessaire afin d'établir la pertinence d'un tel changement d'opérateur.
- l'utilisation du résultat issu du pré–traitement (notamment de l'épuration des points aberrants) afin d'établir la gradualité dans la sortie. A la place d'ignorer les points qui ont été éliminés par l'algorithme d'épuration, on pourrait associer aux régions peuplées par ces points un degré d'appartenance à la classe inférieur à 1, typiquement 0.5. On peut même aller plus loin et

appliquer l'algorithme avec différents seuils, ce qui va partager l'ensemble d'apprentissage en un noyau auquel on associe le degré d'appartenance 1 et en plusieurs couches (qui entourent le noyau) auxquelles on associe des degrés d'appartenance sous–unitaires décroissants.

Enfin, pour augmenter le pouvoir de classification du système proposé on peut envisager la possibilité de définir des formes non–linéaires associées aux classes dans l'espace des attributs. Afin d'aboutir à cette finalité, une étude sur les fonctions d'appartenance utilisées et sur leur impact sur la région désignée pour la classe est nécessaire. L'utilisation des fonctions d'appartenance non–linéaires est envisageable, mais des problèmes d'incohérence peuvent alors apparaître.

Bibliographie

[1] S. Abe, R. Thawonmas, and M. Kayama. A fuzzy classifier with ellipsoidal regions for diagnosis problems. *IEEE Transactions on Systems, Man and Cybernetics – Part C : Applications and Reviews*, 29(1) :140 – 149, 1999.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. *Proc. of the 20th International Conference on Very Large Databases VLDB '94*, 1 :487 – 499, 1994.

[3] P. Antal, G. Fannes, D. Timmerman, Y. Moreau, and B. De Moor. Using literature and data to learn bayesian networks as clinical models of ovarian tumors. *Artificial Intelligence in Medicine*, (30) :257–281, 2004.

[4] T.L. Bailey and C. Elkan. Estimating the accuracy of learned concepts. In *Proc. of the International Joint Conference on Artificial Intelligence 1993 (IJCAI'93)*, pages 895 – 900, Chambéry, France, 1993.

[5] P. Baldi, S. Brunak, Y. Chauvin, C. Andersen, and H. Niels. Assessing the accuracy of prediction algorithms for classification : an overview. *Bioinformatics Review*, 16(5) :412 – 424, 2000.

[6] J.F. Baldwin and N.C.F. Guild. Modelling controllers using fuzzy relations. *Kybernetes*, 9 :223 – 229, 1980.

[7] J. Baron. *Thinking and Deciding*. Cambridge University Press, 2000.

[8] E. Baum. What size net gives valid generalization ? *Neural Computation*, 1(1) :151 – 160, 1989.

[9] A. Bensefia, A. Nosary, T. Paquet, and L. Heutte. Writer identification by writer's invariants. *IEEE Proceedings, 8th International Workshop on Frontiers in Handwriting Recognition*, 1 :274–279, 2002.

[10] J. C. Bezdek. *Pattern Recognition with Fuzzy Objective Function Algoritms*. Plenum Press, New York, 1981.

[11] S. Bhojraj, C.M.C. Lee, and D.K. Oler. What's my line ? A comparison of industry classification schemes for capital market research. *Journal of Accounting Research*, 41(5) :745 – 774, 2003.

[12] I. Bloch. *Applications of Fuzzy Sets Theory*, chapter Dilation and Erosion of Spatial Bipolar Fuzzy Sets, pages 385 – 393. Springer, 2007.

[13] D. Botteldooren, A. Verkeyn, and P. Lercher. Noise annoyance modelling using fuzzy rule based systems. *Noise and Health*, 4(15) :27 – 44, 2002.

[14] O. Brigham. *The Fast Fourier Transform and its Applications*. Prentice-Hall Signal Processing Series, New Jersey, USA, 1988.

[15] D. Bryliuk and V. Starovoitov. Neural networks for access control. *International Workshop and Project Festival , East-West Vision, Graz, Austria*, 1 :203–204, September 2002.

[16] J.J. Buckley and Y. Hayashi. Fuzzy neural networks : a survey. *Fuzzy sets and systems*, 66(1) :1 – 13, 1994.

[17] W.L. Buntine. *A theory of learning classification rules*. PhD thesis, 1992.

[18] W.L. Cameron and L.K. Leung. Feature motivated polarization scattering matrix decomposition. *IEEE Int. Radar Conf.*, pages 549–557, 1990.

[19] G.C. Cawley. Leave-one-out cross-validation based model selection criteria for weighted ls–svms. *Proceedings of the International Joint Conference on Neural Networks (IJCNN – 2006)*, 1 :2970–2977, 2006.

[20] J. Chanussot. *Approches vectorielles ou marginales pour le traitement d'images*. PhD thesis, 1998.

[21] P. Cheeseman, M. Self, J. Kelly, W. Taylor, D. Freeman, and J. Stutz. Bayesian classification. *Seventh National Conference on Artificial Intelligence*, 1 :607–611, 1988.

[22] P. Cheeseman and J. Stutz. Bayesian classification (autoclass) : theory and results. *Advances in knowledge discovery and data mining*, pages 153 – 180, 1996. ISBN : 0-262-56097-6.

[23] Y.W. Choong, L. Di Jorio, A. Laurent, D. Laurent, and M. Teisseire. CBGP : Classification based on gradual patterns. *International Conference of Soft Computing and Pattern Recognition*, 2009.

[24] W. Cohen. Learning rules that classify e-mail. *AAAI Spring Symposium on Machine Learning in Information Access*, pages 18–25, 1996.

[25] S. Colak, D. Papaioannou, G. Hooft, M.V. der Mark, H. Schomberg, J. Paasschens, J. Melissen, and N.V. Asten. Tomographic image reconstruction from optical projections in light–diffusing media. *Applied optics*, 36(1) :180 – 213, 1997.

[26] L. Cooper. The rhetoric of Aristotle. volume 27, pages 694–697, 1932.

[27] T. Cover and P. Hart. Nearest neighbour pattern classification. *IEEE Transactions on Information Theory*, 1 :21 – 27, 1967.

[28] N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines : and other kernel-based learning methods*. Cambridge University Press, New York, USA, 1999.

[29] L. Dârlea, S. Galichet, and L. Valet. Classification rules with premise expressed as the conjunction of implications. *11th Int. Fuzzy Systems Association World Congress (IFSA'2005)*, 2 :1040–1045, 2005.

[30] L. Dârlea, S. Galichet, and L. Valet. Analysis and preprocessing of a learning data set in a cooperative fuzzy classification context. *Rencontres francophones sur la Logique Floue et ses Applications LFA 2006*, pages 53–60, 2006.

[31] L. Dârlea, C. Vertan, M. Ciuc, and I. Stefan. On the influence of image enhancement on fractal-based automatic osteoporosis detection from calcaneum x-rays. *International Symposium on Signals , Circuits and Systems, ISSCS'05*, 1 :43–47, 2005.

[32] B.V. Dasarathy. Sensor fusion potential exploitation-innovative architectures and illustrative applications. *Proceedings of the IEEE*, 85(1) :24 – 28, 2005.

[33] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, (1) :131 – 156, 1997.

[34] J. Daugman. How iris recognition works. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1) :21–30, 2004.

[35] O. Debeir, P. Latinne, and I. Van Den Steen. Remote sensing classification of spectral, spatial and contextual data using multiple classifier systems. *Image Anal Stereol*, (20) :584–589, 2001.

[36] A. Devillez. Four fuzzy supervised classification methods for discriminating classes of non–convex shape. *Fuzzy Sets and Systems*, 141 :219 – 240, 2004.

[37] R. Diestel. *Graph Theory*. Springer–Verlag, New York, USA, 1997, 2000.

[38] T.G. Dietterich. *Neural Computation*, chapter Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms, pages 1895 – 1923. MIT Press, 1998.

[39] M. Donias. *Caractérisation de champs d'orientation par analyse en composantes principales et estimation de la courbure*. PhD thesis, 1999.

[40] D. Dubois, M. Grabish, and H. Prade. Gradual rules and the approximation of control laws. In *Theoretical Aspects of Fuzzy Control*, pages 147 – 181, New York, 1995. Wiley.

[41] D. Dubois and H. Prade. Gradual inference rules in approximate reasoning. *Information Sciences*, 61 :103 – 122, 1992.

[42] D. Dubois and H. Prade. What are fuzzy rules and how to use them. *Fuzzy Sets and Systems*, 84 :169–185, 1996.

[43] D. Dubois, H. Prade, and L. Ughetto. A new perspective on reasoning with fuzzy rules. *International Journal of Intelligent Systems*, 18 :541–567, 2003.

[44] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley & Sons, New York, US, 1973.

[45] S.J. Elliott. Development of a biometric testing protocol for dynamic signature verification. *7th International Conference on Control, Automation, Robotics and Vision ICARCV'02*, 2 :782–787, Dec 2002.

[46] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI Magazine*, 17(3) :37 – 54, 1996.

[47] G.M. Fitzmaurice, W.J. Krzanowski, and D.J. Hand. A monte carlo study of the 632 bootstrap estimator of error rate. *Journal of Classification*, 8 :239 – 250, 1991.

[48] J. D. Forbes. *Travels through the Alps of Savoy and other parts of the Pennin chain with observations on the phenomena of Glaciers*. Adam and Charles Black, Edinburgh, 1845.

[49] A. Freeman and S.L. Durden. A three-component scattering model for polarimetric sar data. *IEEE Transactions on Geoscience and Remote Sensing*, 36(3) :963–973, 1998.

[50] H. Frigui and R. Krishnapuram. Clustering by competitive agglomeration. *Pattern Recognition*, 30(7) :1109 – 1119, 1997.

[51] G. Fumera, F. Roli, and G. Giacinto. Reject option with multiple thresholds. *Pattern Recognition*, 33 :2099 –2101, 2000.

[52] P. Gader, M. Mohamed, and J. Keller. Fusion of handwritten word classifiers. *Pattern Recognition Letters*, 17 :577 – 584, 1996.

[53] S. Galichet, D. Dubois, and H. Prade. Categorizing classes of signals by means of fuzzy gradual rules. *18th Int. Joint Conf. on Artificial Intelligence (IJCAI'03)*, 1 :1039–1044, 2003.

[54] S. Galichet, D. Dubois, and H. Prade. Imprecise specification of ill-known functions using gradual rules. *International Journal of Approx. Reasoning*, 35 :205–222, 2004.

[55] G. Giacinto and F. Roli. Dynamic classifier selection based on multiple classifier behaviour. *Pattern Recognition*, 34 :1879 – 1881, 2001.

[56] M. Grabisch. The application of fuzzy integrals in multicriteria decision making. *European Journal of Operational Re search*, 89 :445 – 456, 1996.

[57] L. Gray and P.J. Farris-Manning. Repeat–pass interferometry with airborne synthetic aperture radar. *IEEE Transactions on Geosience and Remote Sensing*, 31(1) :180 – 191, 1993.

[58] P. Grother. Karhunen Loève feature extraction for neural hanwritten character recognition. In *Proceedings of Applications of Artificial Neural Networks*, volume 1709, pages 155 – 166, SPIE, Orlando, April 1992.

[59] G. Guo and S.Z Li. Content-based audio classification and retrieval by support vector machines. *IEEE Transactions on Neural Networks*, 14(1) :209 – 215, 2003.

[60] B. Guy, M. Daigneault, and G. Thomas. Reflections on the formation of forbes ogives : the instability of fusion of dirty ice. *C. R. Geoscience*, 334 :1061 – 1070, 2002.

[61] J. Han and M. Kamber. *Data Mining : Concepts and Techniques*. Academic Press, USA, 2001.

[62] J.A. Hanley and B.J. McNeil. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143(1) :29–36, 1982.

[63] J.A. Hartigan. *Clustering algorithms*. Wiley series in probability and mathematical statistics, John Wiley, New York, US, 1975.

[64] A. Hauptmann, R. Yan, Y. Qi, R. Jin, M. Christel, M. Derthick, W.-H. Lin M.-Y. Chen, R. Baron, and T. D. Ng. Video classification and retrieval with the informedia digital video library system. *Text Retrieval Conference, Ghaitersburg*, November 2002.

[65] M. Hauta-Kasari, J. Parkkinen, T. Jaaskelainen, and R. Lenz. Generalized co–occurence matrix for multispread texture analysis. *13th International Conference on Pattern Recognition*, 2 :785 – 789, 1996.

[66] S. Haykin. *Neural Networks : A Comprehensive Foundation*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998.

[67] S. Hernandez, D. Mery D. Sz, R. da Silva, and M. Sequeira. Automated defect detection in aluminium castings and welds using neuro-fuzzy classifiers. *Proceedings of 16th World Conference on Non-Destructive Testing (WCNDT 2004)*, August–September 2004.

[68] T.K. Ho, J. Hull, and S. Srihari. Combination of decisions by multiple classifiers. In *Structured Document Image Analysis*, pages 188–202, Springer-Verlag, Heidelberg, 1992. Eds. H. S. Baird and H. Bunke and K. Yamamoto.

[69] F. Hoffmann. Combining boosting and evolutionary algorithms for learning of fuzzy classification rules. *Fuzzy Sets and Systems*, 14 :47 – 58, 2004.

[70] W. Holm and R. M. Barnes. On radar polarization mixed target state decomposition. *IEEE 1998 National Radar Conference*, 1 :249–254, 1998.

[71] H. Hudson and R. Larkin. Accelerated image reconstruction using ordered subsets of projection data. *IEEE Transactions on medical imaging*, 13(4) :601 – 609, 1994.

[72] J. R. Huynen. Measurement of the target scattering matrix. *Proc. IEEE*, 53 :936–946, 1965.

[73] H.Yu, J. Han, and K.C.C. Chang. Pebl : web page classification without negative examples. *IEEE Transactions on Knowledge and Data Engineering*, 16(1) :70 – 81, 2004.

[74] H. Ishibuchi and T. Nakashima. Effect of rule wheights in fuzzy rule–based classification systems. *IEEE Transactions on Fuzzy Systems*, 9(4) :506 – 515, 2001.

[75] H. Ishibuchi, T. Nakashima, and T. Morisawa. Voting in fuzzy rule–based systems for pattern classification problems. *Fuzzy Sets and Systems*, 103 :223–238, 1999.

[76] H. Ishibuchi, K. Nozaki, and H. Tanaka. Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy Sets and Systems*, 52 :21 – 32, 1992.

[77] H. Ishibuchi, K. Nozaki, and H. Tanaka. Efficient fuzzy partition of pattern space for classification problems. *Fuzzy Sets and Systems*, 59 :295 – 304, 1993.

[78] H. Ishibuchi, K. Nozaki, N. Yamamoto, and H. Tanaka. Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms. *Fuzzy Sets and Systems*, 65 :237 – 253, 1994.

[79] H. Ishibuchi and T. Yamamoto. Rule weight specification in fuzzy rule–based classification systems. *IEEE Transactions on Fuzzy Systems*, 13(4) :428 – 435, 2005.

[80] S. Jaillet, A. Laurent, and M. Teisseire. Sequential patterns for text categorization. *Intelligent Data Analysis*, 10(3) :199 – 214, 2006.

[81] A. Jain, R. Duin, and J. Mao. Statistical pattern recognition : A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1) :4 – 37, 2000.

[82] A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering : a review. *ACM Computing Surveys (CSUR)*, 31(3) :264 – 323, 1999.

[83] D. Janssens, G. Wets, T. Brijs, K. Vanhoof, and G. Chen. Adapting the CBA–algorithm by means of intensity of implication. *Information Sciences*, 173(4), 2005.

[84] L. Di Jorio, A. Laurent, and M. Teisseire. Fast extraction of gradual association rules : A heuristic based method. *IEEE/ACM International Conference on Soft Computing as Transdisciplinary Science and Technology (CSTST)*, 1 :205 – 210, 2008.

[85] L. Di Jorio, A. Laurent, and M. Teisseire. Mining frequent gradual itemsets from large databases. *Advances in Intelligent Data Analysis VIII*, 5772 :297 – 308, 2009.

[86] J. Krieter K. Kirchner, K.-H. Tolle. Decision tree technique applied to pig farming datasets. *Livestock Production Science*, (90) :191 – 200, 2004.

[87] L. Kalkstein, G. Tan, and J. Skindlov. An evaluation of three clustering procedures for use in synoptic climatological classification. *Journal of Applied Meteorology*, 26(6) :717 – 730, 1987.

[88] M. Ait Kbir, H. Benkirane, K. Maalmi, and R. Benslimane. Hierarchical fuzzy partition for pattern classification with fuzzy if–then rules. *Pattern Recognition Letters*, 21 :503 – 509, 2000.

[89] P.R. Kersten, J.S. Lee, and T.L. Ainsworth. Unsupervised classification of polarimetric synthetic aperture radar images using fuzzy clustering and em clustering. *IEEE Transactions on Geoscience and Remote Sensing*, 43(3) :519–527, 2005.

[90] G.J. Klir and T.A. Folger. *Fuzzy Sets, Uncertainty and Information*. Prentice–Hall International Edition, San Mateo, California, US, 1988.

[91] R. Kohavi. A study of cross–validation and bootstrap for accuracy estimation and model selection. *International Joint Conference on Artificial Intelligence*, 1 :1137 – 1145, 1995.

[92] B. Kohlmann. History of scarabaeoid classification. *The Coleopterists Bulletin*, 60 :19 – 34, 2006.

[93] E. Krogager. New decomposition of the radar target scattering matrix. *Electronic Letters*, 26(18) :1525–1527, 1990.

[94] L. Kuncheva. A theoretical study on six classifier fusion strategies. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2) :281 – 286, 2002.

[95] L. Kuncheva, J. Bezdek, and R. Duin. Decision templates for multiple classifier fusion : an experimental comparison. *Pattern Recognition*, 34 :299 – 314, 2001.

[96] H.C. Kuo and H.K. Chang. A real-time shipboardfire detection system based on grey-fuzzy algorithms. *Fire Safety Journal*, (38) :341 – 363, 2003.

[97] T. Kölsch, D. Keysers, H Ney, and R. Paredes. Enhancements for local feature based image classification. *ICPR*, (1) :248–251, 2004.

[98] A. Laurent, C. Marsala, and B. Bouchon-Meunier. Improvement of the interpretability of fuzzy rule based systems : quantifiers, similarities and aggregators. In *Modelling with words, series 'Lecture Notes in Artificial Intelligence', LNAI 2873*, pages 102 – 123. Springer-Verlag, A. Ralescu, J. Lawry, J. Shanahan, 2003.

[99] N. Lavra, P. Flach, and B. Zupan. *Inductive Logic Programming*, chapter Rule Evaluation Measures : A Unifying View, pages 174 – 185. Springer, 1999.

[100] A. Leouski and J. Allan. *Research and Advanced Technology for Digital Libraries*, chapter Evaluating a Visual Navigation System for a Digital Library, pages 535 – 554. Springer Berlin / Heidelberg, 1998.

[101] B. Lerner and N. D. Lawrence. A comparison of state-of-the-art classification techniques with application to cytogenetics. *Neural Computations and Applications*, (10) :39 – 47, 2001.

[102] W. Li, J. Han, and J. Pei. CMAR : Accurate and efficent classification based on multiple class–association rules. *Proc. 2001 of International Conference on Data Mining ICDM '01*, 1 :369 – 376, 2001.

[103] S. Ee Lim, Y. Xing, Y. Chen, W. Kheng Leow, T. Sen Howe, and M. Ai Png. Detection of femur and radius fractures in X-ray images. *Proceedings of 2nd International Conference on Advances in Medical Signal and Information Processing*, 2004.

[104] J. Lin and L. Qu. Feature extraction based on Morlet wavelet and its application for mechanical fault diagnosis. *Journal of Sound and Vibration*, 1(234) :135 – 148, 2000.

[105] B. Liu, W. Hsu, and Y. Ma. Integrating classification and association rule mining. *Knowledge Discovery and Data Mining*, 1 :80 – 86, 1998.

[106] J. Liu, H. Iba, and M. Ishizuka. Selecting informative genes with parallel genetic algorithms in tissue classification. *Genome Informatics*, (12) :14 – 23, 2001.

[107] L. Lliboutry and L. Reynaud. Global dynamics of a temperate valley glacier, mer de glace, and past velocities deduced from forbes' bands. *Journal of Glaciology*, 27(96) :207 – 226, 1981.

[108] C. Lu, T. Van Gestel, J.A.K. Suykens, S. Van Huffel, I. Vergote, and D. Timmerman. Preoperative prediction of malignancy of ovarian tumors using least squares support vector machines. *Artificial Intelligence in Medicine*, (28) :281–306, 2003.

[109] D. Maltoni, D. Maio, A.K. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition*. Springer, USA, 2003.

[110] J. Manuel, F. Salido, and S. Murakami. A comparison of two learning mechanisms for the automatic design of fuzzy diagnosis systems for rotating machinery. *Applied Soft Computing*, (4) :413–422, 2004.

[111] G. Marinescu, L. Valet, P. Lambert, and S. Teyssier. Electrotechnical parts quality control by computed tomography. *IEEE Intrumentation and Measurement Technology Conference*, 2 :1444–1448, 2004.

[112] I.W. McKeague. A statistical model for signature verification. *Journal of the American Statistical Association*, 100 :231–241, 2005.

[113] D. Meretakis and B. Wuthrich. Extending naive Bayes classifiers using long itemsets. *Knowledge Discovery and Data Mining*, 1 :165 – 174, 1999.

[114] C. Metz, B. Herman, and J-H. Shen. Maximum likelihood estimation of receiver operating characteristic (ROC) curves from continuously–distributed data. *Statistics in Medicine*, 17 :1033 – 1053, 1998.

[115] B. Moghaddam, T. Jebara, and A. Pentland. Bayesian face recognition. *Pattern Recognition*, 33(11) :1771–1782, November 2000.

[116] D. Moshou, C. Bravo, J. West, S. Wahlen, A. McCartney, and H. Ramon. Automatic detection of "yellow rust" in wheat using reflectance measurements and neural networks. *Computers and Electronics in Agriculture*, (44) :173 – 188, 2004.

[117] D. Nauck. Measuring interpretability in rule-based classification systems. *The 12th IEEE International Conference on Fuzzy Systems FUZZ '03*, 1 :196 – 201, 2003.

[118] M. Paik and Y. Yang. Combining nearest neighbor classifiers versus cross-validation selection. *Statistical Applications in Genetics and Molecular Biology*, 3, 2004.

[119] S.K. Pal and S. Mitra. Multilayer perceptron, fuzzy sets, and classification. *IEEE Transactions on Neural Networks*, 3(5) :683 – 697, August 2002.

[120] J.R. Parker. Rank and response combination from confusion matrix data. *Information Fusion*, 2 :113 – 120, 2001.

[121] A. Pentland and T. Choudhury. Face recognition for smart environments. *Computer Science*, 33(2) :50–55, 2000.

[122] D. Pietersma, R. Lacroix, D. Lefebvre, and K.M. Wade. Induction and evaluation of decision trees for lactation curve analysis. *Computers and Electronics in Agriculture*, (38) :19–32, 2003.

[123] E. Pottier, L. Ferro-Famil, S. Cloude, I. Hajnek, K. Papathanassion, A. Moreira, T. Pearson, and Y-L. Desnos. Polsarpro v2.0 software. a versatile polarimetric sar data processing and educational toolbox. *Proceedings of POLinSAR 2005 Symposion*, 2005. ISBN : 92-9092-897-2.

[124] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1 :81 – 106, 1986.

[125] K. Venkat Ramana and B. Ramamoorthy. Statistical methods to compare the texture features of machined surfaces. *Pattern Recognition*, 29(9) :1447 – 1459, 1996.

[126] S. Ramasso-Jullien. *Systèmes coopératifs explicitant les dépendances entre les informations : application à l'interprétation d'images tomographiques 3D et à la sélection de films d'animation.* PhD thesis, 2008.

[127] V. Ravi, P.J. Reddy, and H.J. Zimmermann. Pattern classification with principal component analysis and fuzzy rule bases. *European Journal of Operational Research*, (126) :526 – 533, 2000.

[128] V. Ravi, P.J. Reddy, and H.J. Zimmermann. Fuzzy rule base generation for classification and its minimization via modified threshold accepting. *Fuzzy Sets and Systems*, (120) :271 – 279, 2001.

[129] F. Del Razo, A. Laurent, P. Poncelet, and M. Teisseire. FTM nodes : fuzzy tree mining based on partial inclusion. *Fuzzy Sets and Systems*, Special Issue : The Application of Fuzzy Logic and Soft Computing in Information Management :2224 – 2240, 2009.

[130] A. Reigber and R. Scheiber. Differential SAR interferometry using an airborne platform. *Proceedings of EUSAR'02*, pages 373 – 376, 2002.

[131] P. Rich. The organizational taxonomy : Definition and design. *Academy of Management Review*, 4 :758 –781, 1992.

[132] I. Rish. An empirical study of the naive bayes classifier. *Proceedings of IJCAI-01 workshop on Empirical Methos in Artificial Intelligence*, pages 41 – 46, 2001.

[133] A. Rodrigues, D. Corr, K. Partington, E Pottier, and L. Ferro-Famil. Unsupervised wishart classification of sea-ice using entropy, alpha and anisotropy decompositions. *Proceedings of the Workshop on POLinSAR-Applications of SAR*, 2003.

[134] F. Rossi, B. Conan-Guezand, and A. El Golli. Clustering functional data with the SOM algorithm. *European Symposion on Artificial Neural Networks Proceedings (ESANN'04)*, 2004.

[135] J.A. Roubos, M. Setnes, and J. Abonyi. Learning fuzzy classification rules from labeled data. *Information Sciences*, (150) :77 – 93, 2003.

[136] D. Ruta and B. Gabrys. An overview of classifier fusion methods. *Computer and Information Systems*, 7 :1 –10, 2000.

[137] F. Rémy and L. Testut. But how a glacier can flow ? Historical outlines. *C. R. Geoscience*, 338 :368 – 385, 2006.

[138] J. Sanchez and L.I. Kuncheva. Data reduction using classifier ensembles. *Proc. 11th European Symposium on Artificial Neural Networks*, pages 379 – 384, 2007. ISBN : 2-930307-07-2.

[139] P. Shaoning, S. Ozawa, and N. Kasabov. Incremental linear discriminant analysis for classification of data streams. *IEEE Transactions on systems, man, and ybernetics. Part B, Cybernetics*, 35(5) :905 – 914, 2005.

[140] A. Sharkey and N. Sharkey. Combining diverse neural nets. *The Knowledge Engineering Review*, 12(3) :231 – 247, 1997.

[141] B. Shen and K. Sethi. Direct feature extraction from compressed images. *SPIE – Storage and Retrieval for Image and Video Databases*, 2670.

[142] Q. Shen and A. Chouchoulas. A rough–fuzzy approach for generating classification rules. *Pattern Recognition*, (35) :2425 – 2438, 2002.

[143] M. Sokolova. Assessing invariance properties of evaluation measures. *Proc. of the Workshop on Testing of Deployable Learning and Decision Systems, 19th Neural Information Processing Systems Conference (NIPS 2006)*, 2006.

[144] V.V. Starovoitov, D.I Samal, and D.V. Briliuk. Three approaches for face recognition. *6-th International Conference on Pattern Recognition and Image Analysis, Velikiy Novgorod, Russia*, 1 :707–711, October 2002.

[145] A. Statnikov, C.F. Aliferis, I. Tsamardinos, D. Hardin, and S. Levy. A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics Advance Access*, 21(5) :631 – 643, September 2005.

[146] P. Subasic and K. Hirota. Similarity rules and gradual rules for analogical interpolative reasoning with imprecise data. *Fuzzy Sets and Systems*, 96 :53–75, 1998.

[147] S. Teyssier. *Méthodologie de caractérisation de l'architecture 3D et multiéchelle de composites renforcés par des fibres.* PhD thesis, 2004.

[148] S. Thrun, C. Faloutsos, T. Mitchell, and L. Wasserman. Automated learning and discovery : State of the art and research topics in a rapidly growing field. *AI MAG*, 20(3) :78–82, June 1999.

[149] N. Ueda and R. Nakano. SMEM algorithm for mixture models. *Neural Computation*, 12 :2109 – 2128, 2000.

[150] L. Valet, G. Mauris, P. Bolon, and N. Keskes. A fuzzy rule-based interactive fusion system for seismic data analysis. *Information Fusion*, 4(2) :123 – 133, June 2003.

[151] V. Vapnik, E. Levin, and Y.L. Cun. Measuring the vc–dimension of a learning machine. *Neural Computation*, 6(5) :851 – 876, 1994.

[152] C. Vertan, B. Ionescu, I. Stefan, and M. Ciuc. Fractal and second order statistics for the calcaneum trabecular structure analysis. *14th International Conference on Control Systems and Computer Science (CSCS)*, 2 :281–284, July 2003.

[153] H. vonHelmholtz. *Ice and glaciers. Scientific Papers. The Harvard Classics.* P. F. Collier and Son, New York, 2001.

[154] S. Watanabe. *Pattern Recognition : Human and Mechanical*. Wiley, New York, 1985.

[155] S.M. Weiss and C.A. Kulikowski. *Computer Systems that Learn : Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning and Expert Systems*. Morgan Kaufmann Publishers, San Mateo, California, US, 1990.

[156] R.P. Wildes. Iris recognition : An emerging biometric technology. *Proceedings of the IEEE*, 85(9), September 1997.

[157] K. Woods, W.P. Kegelmeyer, and K. Bowyer. Combination of multiple classifiers using local accuracy estimates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4) :405–410, 1997.

[158] L. Xu and A. Krzyzak. Methods of combining multiple classifiers and their applications to handwritting recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 22(3) :418 – 435, 1992.

[159] L. A. Zadeh. Fuzzy sets. *Information and Control*, 8 :338 – 353, 1965.

[160] Y. Zimmer, R. Tepper, and S. Akselrod. An automatic approach for morphological analysis and malignancy evaluation of ovarian masses using b-scans. *Ultrasound in Medicine and Biology*, 29(11) :1561–1570, 2003.

Résumé

Le travail de recherche de la thèse concerne la classification supervisée de données et plus particulièrement l'apprentissage semi–automatique de classifieurs à base de règles floues graduelles. Le manuscrit de la thèse présente une description de la problématique de classification ainsi que les principales méthodes de classification déjà développées, afin de placer la méthode proposée dans le contexte général de la spécialité. Ensuite, les travaux de la thèse sont présentés : la définition d'un cadre formel pour la représentation d'un classifieur élémentaire à base de règles floues graduelles dans un espace 2D, la spécification d'un algorithme d'apprentissage de classifieurs élémentaires à partir de données, la conception d'un système multi–dimensionel de classification multi–classes par combinaison de classifieurs élémentaires. L'implémentation de l'ensemble des fonctionnalités est ensuite détaillée, puis finalement les développements réalisés sont utilisés pour deux applications en imagerie : analyse de la qualité des produits industriels par tomographie, classification en régions d'intérêt d'images satellitaires radar.

Abstract

This PhD thesis presents a series of research works done in the field of supervised data classification, more precisely in the domain of semi–automatic learning of fuzzy rules–based classifiers. The prepared manuscript presents first an overview of the classification problem, and also of the main classification methods that have already been implemented and certified in order to place the proposed method in the general context of the domain. Once the context established, the actual research work is presented : the definition of a formal background for representing an elementary fuzzy rule-based classifier in a bidimensional space, the description of a learning algorithm for these elementary classifiers for a given data set and the conception of a multi-dimensional classification system which is able to handle multi–classes problems by combining the elementary classifiers. The implementation and testing of all these functionalities and finaly the application of the resulted classifier on two real–world digital image problems are finaly presented : the analysis of the quality of industrial products using 3D tomographic images and the identification of regions of interest in radar satellite images.